PROGRAMAS PRACTICOS PARA EL AMSTRAD

UNA BIBLIOTECA DE MODULOS Y SUBRUTINAS

D. LAWRENCE / S. LANE

AMSTRAD

PROGRAMAS PRACTICOS PARA EL AMSTRAD

Editorial Gustavo Gili, S. A.

08029 Barcelona Rosellón, 87-89. Tel. 322 81 61
28006 Madrid Alcántara, 21. Tel. 401 17 02
1064 Buenos Aires Cochabamba, 154-158. Tel. 361 99 98
México, Naucalpan 53050 Valle de Bravo, 21 - Tel. 560 60 11
Bogotá Diagonal 45 N.° 16 B-11. Tel. 245 67 60
Santiago de Chile Vicuña Mackenna, 462, Tel. 222 45 67

PROGRAMAS PRACTICOS PARA EL AMSTRAD

UNA BIBLIOTECA DE MODULOS Y SUBRUTINAS

D. LAWRENCE / S. LANE

GG

Título original
The Working Amstrad
A library of practical subroutines and programs

Publicado originalmente en inglés en 1984 por:
Sunshine Books (an imprint of Scot Books Ltd.)
12/13 Little Newport Street
London WC2H 7PP.

Versión castellana de Luis Alegre Elvira, licenciado en Ciencias Físicas

Printed in Spain
ISBN: 84-252-1288-X
Depósito legal: B. 20.624-1986
Fotocomposición: TECFA, S. A. - Barcelona
Impresión: Gráficas 92 - San Adrián del Besós

Indice

El contenido en detalle

1. Experimentos horarios
Analoj: presenta un reloj, de esfera tradicional, en alta resolución (se aprende a definir un círculo) —Reloj: proporciona una forma totalmente diferente de expresar la hora —Temporizador: le proporciona 16 temporizadores que corren simultáneamente, cada uno de los cuales es capaz de hacer sonar una alarma y de mostrar un mensaje recordatorio —Cronómetro: convierte a su CPC464 en un cronómetro sofisticado.

2. Pintando con números
Gráficos: crea gráficos lineales muy eficaces y potentes en alta resolución —Gráfico-sectores: tomará una cantidad finita de datos y los mostrará bajo la forma de un círculo multicolor dividido en sectores circulares —Gráficos-3D: le permitirá crear un sorprendente gráfico de barras en tres dimensiones (3D).

3. Son et lumière
Caracteres: permite crear juegos de caracteres a gusto del usuario (memoria de caracteres) —Diseñador: una herramienta para crear diseños en alta resolución —Música: le permite introducir melodías a tres voces, en un formato sencillo, y oírlas cómo suenan.

4. Más y más serio
Uniarchivo: un potente sistema de fichero personal capaz de almacenar una amplia variedad de información para su inmediata recuperación (búsqueda binaria) —Nnúmero: crea un diccionario de nombres y números para utilizar en casi cualquier asunto que desee; permite al usuario la expedición de facturas, la estimación de existencias o incluso el recuento de las calorías del menú del día —Editexto: un sencillo paquete para procesamiento de palabras —Multirrespuesta: un generador de preguntas a las que se le asocian varias respuestas, de las cuales sólo una es la correcta.

5. Cuestiones monetarias
Banquero: guarda un registro ordenado de los ingresos y pagos que se realizan en una cuenta bancaria —Contable: compila una serie de cuentas en formato tradicional.

Notas de programación

Ordenes de color

Todos los programas de este libro fueron diseñados y comprobados sobre un Amstrad estándar con monitor de color. Aquellos lectores que posean un monitor monocromático (verde sobre negro) encontrarán de utilidad modificar las órdenes de color en algunas de las etapas.

Almacenamiento de datos

El almacenamiento de datos se efectúa por medio del Datacorder (casete) incorporado al propio CPC464. Aquellos lectores que posteriormente compren una unidad de disco compatible, encontrarán muy poca dificultad en alterar las órdenes de apertura y cierre de ficheros, para que se adapten a su sistema ampliado.

Signo de exponenciación

El símbolo ^ que aparece en los programas es el símbolo de flecha hacia arriba (↑), el signo de exponenciación.

Introducción

Este libro forma parte de la serie de más éxito de libros sobre microordenadores que jamás se haya publicado. En 1982, cuando se lanzó el primer libro «Programas prácticos para el...», muy pocos editores estaban preparados para creer que los propietarios normales de micros deseaban utilizar y comprender sus máquinas, que querían poder *controlarlas*, aprendiendo a programar (programando). La mayoría de los libros estaban repletos de juegos y trivialidades, o no eran más que otra *Guía para «principiantes» para el...* La perspectiva de un libro que se proponía suministrar una colección de programas completos y útiles, y que al mismo tiempo daba una ligera visión sobre los métodos empleados en programación seria, no parecía nada probable que fuera a cubrirse de gloria.

Sin embargo, yo estaba convencido, al igual que los componentes de Sunshine Books, de que libros tipo «Programas prácticos para el...» eran exactamente lo que la gente andaba buscando, de que existía un enorme vacío dentro de los libros publicados para el creciente ejército de propietarios de micros. Y estábamos en lo cierto.

A partir de entonces, los libros «Programas prácticos para el...» han seguido a los microordenadores a casi todos los países en donde éstos se han vendido; y han sido, o lo están siendo, traducidos a 14 idiomas.

Ahora llega desde Amstrad una nueva y audaz máquina, el CPC464, con una versión nueva y brillante de BASIC, una memoria más que suficiente para casi cualquier propósito, junto con una amplia gama de recursos de explotación sin paralelo en lo que a su extraordinariamente bajo precio se refiere. El 464 y la idea que subyace tras los libros «Programas prácticos para el...» están hechos el uno para el otro, cosa que esperamos le quede probada por los programas que aparecen en este libro.

Cómo utilizar este libro

Este libro puede utilizarse de varias formas diferentes:

1) Como una colección de programas útiles que puede adaptar y desarrollar para sus propios objetivos.

2) Como una colección de subrutinas a partir de las cuales puede construirse sus propios programas.
3) Como una introducción a la programación, utilizando el BASIC del CPC464.

Sea cual fuere la forma en que decida utilizarlo, recuerde que fue escrito *en forma de libro* y no en forma de colección aleatoria de programas sin orden alguno. Con mucha frecuencia nos encontramos con lectores que tienen problemas a causa de haberse saltado, sin ninguna preparación, a alguno de los programas más complejos que hay hacia el final del libro. Los primeros programas del libro, aun cuando útiles e interesantes en sí mismos, también están pensados como introducción para los posteriores. Estos primeros programas van acompañados de un nivel de explicaciones y aclaraciones mucho más alto, de forma que cuando se llega a programas más complejos el lector tiene un buen dominio de algunas de las técnicas que se están utilizando.

1. Experimentos horarios

Siempre resulta difícil saber el nivel con que comenzar un libro de este tipo: un programa demasiado complejo y es posible que algunos lectores se encuentren atascados sin haber aprendido algunas de las simples indicaciones que harán que los programas sean progresivamente más fáciles de entender conforme se avanza por el libro; por otra parte, si los primeros programas son demasiado triviales, puede que muchos lectores no se preocupen por descubrir que hay cosas más sustanciales en los restantes.

En consecuencia, he decidido poner en este primer capítulo un conjunto de cuatro programas que tratan sobre relojes y contadores, y sobre la forma en que pueden ser manipulados con el CPC464. Los programas son relativamente sencillos, pero introducen un amplio conjunto de conceptos que se utilizarán en programas posteriores de mayor complejidad. Además, el uso que se hace de los sonidos, gráficos y cálculos, proporcionará una buena introducción a alguna de las capacidades más sobresalientes de su CPC464.

Los programas que se presentan en este capítulo son:

ANALOJ: Que presenta un reloj, de esfera tradicional, en alta resolución.

RELOJ: Que proporciona una forma totalmente diferente de expresar la hora.

TEMPORIZADOR: Que le proporciona 16 temporizadores que corren simultáneamente, cada uno de los cuales es capaz de hacer sonar una alarma y de mostrar un mensaje recordatorio.

CRONOMETRO: Que convierte a su CPC464 en un cronómetro sofisticado (y bastante caro).

PROGRAMA 1.1: Analoj

Función del programa

El objetivo de este programa es generar, en pantalla, una réplica de la esfera de un reloj completada con sus manecillas, manecillas que deberán moverse acompasadamente con el tiempo real. Durante el seguimiento del programa aprenderá bastante sobre los métodos empleados en este libro, por lo que se recomienda que lea atentamente los comentarios adjuntos.

Fig. 1.1 Vaciado de pantalla del programa Analoj. El aspecto ligeramente aplastado es consecuencia del proceso de vaciado de pantalla (la esfera del reloj es circular cuando se presenta en pantalla).

Las ideas introducidas en el transcurso del programa incluyen:

1) Conservación (grabación) de los programas durante su desarrollo.
2) Inicialización del programa.
3) Módulos de control.
4) Sincronización y/o temporización mediante la orden EVERY.
5) Programación modular.
6) El módulo de control.
7) Las sencillas matemáticas que definen una circunferencia.

Módulo 1.1.1: Conservación del programa

Estas tres líneas pueden parecer un comienzo insignificante, pero aquellos que han manejado los libros «Programas prácticos para el...» con otros micros saben que este pequeño módulo puede evitarnos ingentes dosis de desesperación en el desarrollo de los programas.

La mayoría de las personas aprenden solamente, tras amargas experiencias, que los programas *deben* ser grabados regularmente, es decir, a lo largo de su desarrollo. Antes o después, a la mayoría de nosotros nos llega el instante fatal en que de repente perdemos horas de trabajo por culpa de un transitorio en la alimentación, de un fusible que se funde o de un golpe al micro o a un conector. Los usuarios experimentados sólo habrán perdido unos 15 minutos de trabajo, simplemente porque jamás dejan que pasen más de 15 minutos sin guardar (mediante SAVE) el programa introducido hasta ese momento.

El propósito de las tres líneas de este módulo es incitarle a que haga copias frecuentes, sin más que teclear GOTO 2, del programa en el que está trabajando. Otro aspecto que le ahorrará tiempo en el futuro es que un módulo como éste, colocado al comienzo de todos sus programas, le proporciona una línea estándar de arranque para todos ellos: la «1». Con mucha frecuencia es deseable comenzar un programa con un GOTO, por ejemplo cuando las variables ya se han ajustado y no se desea que se borren como ocurriría si utilizamos un RUN (con este módulo no tiene que recordar el número de la primera línea de un programa particular, sabe que siempre podrá arrancar con un GOTO 1).

Conforme se fueron desarrollando los programas del libro este módulo precedía a todos ellos, pero ya que sólo cambia el nombre del programa no volverá a incluirse en ninguno de los programas restantes listados en este libro.

Módulo 1.1.1: Líneas 1 - 3

```
1 GOTO 3
2 SAVE «analoj»:STOP
3 REM
```

Comprobación

Asegúrese de que tiene un cinta en el casete y teclee:

```
GOTO 2[ENTER]
```

Siga las instrucciones del sistema para poner en marcha el gra-

bador. Al poco rato debería ver BREAK IN 2 (RUPTURA EN LINEA 2) en la pantalla. Ahora puede borrar las tres líneas que hay en memoria, y puede cargar el programa desde el casete sin más que rebobinar la cinta y teclear:

NEW[ENTER] (lo que hace que se borre el programa actual)
LOAD «ANALOJ»[ENTER]

Cuando haya finalizado el proceso de carga, liste el programa y observará que el módulo debe haber sido restituido.

Módulo 1.1.2: Inicialización del programa

Cualquier programa merecedor de ese nombre utiliza variables y constantes, es decir: etiquetas cuyos valores pueden ser cambiados durante el curso del programa, o al menos de programa a programa. La ventaja de las variables es simplemente que le permiten escribir líneas de programa que se aplican a más de una situación; por ejemplo, podría utilizarse PRINT 2*A sin que importe el valor de A. Muy pocas variables *deben* tener sus valores declarados cuando el programa es ejecutado (RUN) por primera vez, y la gente suele dejar para mitad del programa, cuando una variable es absolutamente vital, la definición de sus valores. Esto es una táctica engañosa ya que, conforme se va desarrollando el programa, se va haciendo más difícil ver cuál es el valor de las variables importantes cuando el programa comienza por vez primera. En general, es una buena práctica declarar el valor de las variables principales justo al principio del programa, proceso al que se le conoce como «inicialización».

Una excepción sensata, cuando tenemos una memoria limitada, es omitir en dicha inicialización aquellas variables cuyo valor *carece de importancia* cuando el programa comienza por vez primera. Así, aparte de los valores que representan los colores que se van a utilizar, todas las variables importantes que se utilizan en este programa se derivan de lo que introduzca el usuario cuando se le pide que especifique la hora; por lo tanto, tales variables, que son irrelevantes cuando el programa comienza por vez primera, no se declararán en el módulo de inicialización.

En el caso de este programa en particular, el objeto del módulo es ajustar los parámetros de la presentación gráfica final, y poner la máquina en modo «grados» (DEGree), lo que es muy conveniente en cálculos que implican informaciones horarias.

Módulo 1.1.2: Líneas 2000 - 2110

```
2000 REM*******************************
2010 REM Inicializacion
2020 REM*******************************
2030 MODE 1
2040 INK 0,1
2050 INK 1,24
2060 INK 2,3
2070 INK 3,6
2080 BORDER 1
2090 ORIGIN 200,200
2100 DEG
2110 RETURN
```

Comprobación

Se puede hacer una comprobación tosca del módulo sin más que introducir:

```
GOTO 2000
```

Si el módulo se ha introducido correctamente se limpiará la pantalla y aparecerá el mensaje «Unexpected RETURN in 2110» (RETURN inesperado en línea 2110). La comprobación total del funcionamiento del módulo solamente será posible cuando se haya introducido el resto de los módulos.

Módulo 1.1.3: Ajuste inicial de la hora

Antes de emprender el diseño gráfico del reloj debemos tener algún medio para ajustar la hora. El objetivo de este módulo es permitir al usuario que introduzca la hora real (en horas y minutos) y almacenarla en la memoria del 464 en dos variables: HORA y MINUTO.

Módulo 1.1.3: Líneas 3000-3080

```
3000 REM*******************************
3010 REM Ajuste de la hora
3020 REM*******************************
3030 PRINT "Ajuste la hora:":PRINT
3040 INPUT "Hora (1 a 12):";hora
3050 IF hora<1 OR hora>12 THEN PRINT "**
*Fuera del intervalo: introduzca dato ot
ra vez***":GOTO 3040
```

```
3060 INPUT "minutos (0 a 59):";minuto
3070 IF minuto<0 OR minuto>59 THEN PRINT
 "***Fuera del intervalo: introduzca dat
o otra vez***":GOTO 3060
3080 RETURN
```

Comprobación

Introduzca:

GOTO 3000[ENTER]

e inmediatamente se le pedirá que exprese la hora actual en horas y minutos. Si introduce una cifra ilógica (ya sea en las horas o en los minutos) aparecerá un mensaje de error, en tanto que será aceptado cualquier valor que sea lógico. Cuando se haya aceptado una hora, el programa se detendrá con un mensaje de error: «Unexpected RETURN». No se preocupe, no hay nada mal; una vez se haya introducido el módulo final, este otro módulo será una subrutina a la que se llamará mediante un GOSUB.

Si lo desea, puede hacer una comprobación adicional tecleando:

PRINT hora,minuto[ENTER]

lo que debería dar lugar a que salgan en pantalla los valores de la hora y los minutos que introdujo anteriormente.

Módulo 1.1.4: Trazado de la esfera del reloj

Una vez introducida la hora pasemos a dibujar la esfera del reloj, sobre la que módulos posteriores colocarán las manillas.

Módulo 1.1.4: Líneas 4000 - 4090

```
4000 REM*******************************
4010 REM Esfera del reloj
4020 REM*******************************
4030 CLS
4040 FOR a=0 TO 359 STEP 6
4050 MOVE 200*SIN(a),200*COS(a)
4060 r=195:IF a MOD 30=0 THEN r=185
4070 DRAW r*SIN(a),r*COS(a),1
4080 NEXT a
4090 RETURN
```

Comentarios

Definición de un círculo

El método empleado en este módulo se basa en el hecho de que puede determinarse cualquier punto perteneciente a la circunferencia de un círculo si se conocen los datos siguientes:

a) El radio del círculo (RADIO).
b) El ángulo que debe (recorrerse en sentido horario) desde la posición de las tres en punto hasta llegar al punto especificado (ANGULO).
c) Las coordenadas del centro del círculo (X CENTRO e Y CENTRO).

Dados estos tres datos, la posición del punto puede calcularse mediante las dos fórmulas:

Coordenada X=RADIO*COSENO(ANGULO)+X CENTRO
Coordenada Y=RADIO*SENO(ANGULO)+Y CENTRO

El espacio de que disponemos no nos permite analizar porqué esto es así, pero cualquier buen libro introductorio de trigonometría lo explicará claramente. En nuestro caso particular las fórmulas son aún más sencillas, ya que una de las cosas que hicimos en el módulo de inicialización fue trasladar el origen (ORIGIN) de gráficos de la pantalla al punto de coordenadas (200,200). Esto significa que cualquier referencia a la posición (0,0) aparecerá en pantalla en una posición que está 200 pixels más arriba y más a la derecha que la esquina inferior izquierda de dicha pantalla. La ventaja de esto es que, habiendo movido el origen de gráficos al lugar donde queremos centrar la esfera del reloj, podemos omitir cualquier referencia a las coordenadas X e Y del centro de tal esfera (ya que ahora, ambas son cero).

Por tanto, las fórmulas que dan ahora la posición de cualquier punto de la periferia de un círculo son:

Coordenada X=RADIO*COSENO(ANGULO)
Coordenada Y=RADIO*SENO(ANGULO)

que, como puede ver, son exactamente las que se han utilizado en el módulo.

Líneas 4040-4080: Este bucle nos hace recorrer los 360 grados de la circunferencia en saltos de 6 grados. Puesto que una hora tiene 60 minutos, y 360/60=6, no se sorprenderá al saber que el bucle tiene que ver con las marcas de los minutos de la esfera, marcas que serán dibujadas por las líneas siguientes.

Línea 4050: Mediante las fórmulas descritas anteriormente se mueve (MOVE) el cursor de gráficos a un punto de la circunferencia. La primera posición (cuando la variable A del bucle es cero) está en la parte superior de la circunferencia, y las siguientes posiciones la recorren en sentido horario y en incrementos de 6 grados.
Líneas 4060-4070: Una vez establecida una posición en la periferia del círculo, lo que hacen estas líneas es dibujar una raya hacia el centro. En realidad, lo que hacen es dibujar una raya desde la periferia del círculo hasta la periferia de otro círculo más pequeño contenido en el primero. El círculo exterior tiene un radio de 200 pixels, mientras que el del interior depende de si el ángulo indicado por la variable A del bucle es múltiplo entero (o no) de 30 grados (30 grados corresponden a 5 minutos). En caso de que no sea múltiplo entero, el radio del círculo interior tendrá solamente cinco pixels menos que el del exterior; es decir, la raya dibujada tendrá una longitud de cinco pixels.

El uso de la función MOD asegura que el programa reconoce cuándo se alcanza un ángulo múltiplo de 30 grados. Por ejemplo, el resultado de 5 MOD 2 es 1, o sea lo que sobra (el resto) al dividir 5 entre 2. Del mismo modo, A MOD 30 (la expresión que aparece en el programa) da como resultado el resto de la división de la variable A del bucle entre 30, por lo que cuando el resultado sea cero querrá decir que el ángulo es divisible exactamente por 30. Cada vez que ocurre esto, el círculo interior se hace más pequeño, o lo que es lo mismo la raya dibujada se hace más larga, resaltándose así las marcas correspondientes a los múltiplos de cinco minutos.

Comprobación

Teclee:

```
GOTO 4000[ENTER]
```

y debería ver cómo se limpia la pantalla y se dibuja la esfera de un reloj. El programa terminará entonces con un mensaje de error: «Unexpected RETURN».

Módulo 1.1.5: Ajuste del minutero y de la aguja horaria

En este módulo abordamos el trabajo real del programa, que no es más que calcular la hora actual y, basados en ella, las coordenadas de las manillas del reloj. El módulo no se utilizará correctamente hasta que no se haya introducido el último de todos, ya que solamente entonces será llamado en los instantes adecuados, es decir: una vez cada minuto.

```
5000 REM*******************************
5010 REM Ajustes horarios
5020 REM*******************************
5030 c=0
5040 angulo=minuto*6:longitud=180:GOSUB
6000
5050 angulo=hora*30+minuto/2:longitud=12
0:GOSUB 6000
5060 minuto=minuto+1
5070 IF minuto=60 THEN minuto=0:hora=hor
a+1
5080 IF hora=13 THEN hora=1
5090 c=2
5100 angulo=minuto*6:longitud=180:GOSUB
6000
5110 c=3
5120 angulo=hora*30+minuto/2:longitud=12
0:GOSUB 6000
5130 RETURN
```

Comentarios

Realmente el módulo se divide en tres secciones diferentes. La tarea de la primera sección es llamar a un módulo posterior para que borre las manillas de su posición actual. La segunda sección suma 1 a la variable MINUTO, y en caso necesario realiza los cambios pertinentes en la variable HORA. La tercera sección toma los nuevos valores hora/minuto y llama a un módulo posterior para que dibuje las manillas en su nueva posición.

Líneas 5030-5050: La variable C se utilizará por un módulo posterior para fijar el color de la tinta cuando se dibujen las manillas. Primero se fija al valor 0. Si retrocede al módulo de inicialización verá que el color 0 (el color de fondo) se definió como azul. La línea 5040 ajusta los parámetros del minutero, a saber: el ángulo a que debe ser trazado y su longitud. Puesto que el valor de los minutos aún no se ha cambiado, dibujar el minutero con el color de fondo tiene el efecto de borrarlo. La línea 5050 hace exactamente lo mismo, sólo que con la manilla de las horas.

Líneas 5060-5080: Se actualizan los valores de los minutos y de las horas, para ello se suma 1 a los minutos y se hacen los ajustes necesarios para asegurar que los valores resultantes sean los lógicos.

Líneas 5090-5120: Antes de llamar al módulo siguiente para que vuelva a dibujar las manillas, se ajustan de nuevo los parámetros de ambas, utilizando para ello los valores actualizados. El número de co-

lor C se pone a 2 para el minutero, y a 3 para la manilla horaria. Retrocediendo al módulo de inicialización encontrará que estos valores se corresponden con el rojo y el rojo brillante respectivamente.

Comprobación

Teclee:

6000 RETURN[ENTER]

que no es más que una línea provisional para tener en cuenta el hecho de que el módulo que acaba de introducir llama a otro que aún no existe.

Ahora teclee:

HORA=0[ENTER]
MINUTO=59[ENTER]
GOTO 5000[ENTER]

y debería observar, casi instantáneamente, que se interrumpe la ejecución con un mensaje de error «Unexpected RETURN». Teclee ahora:

PRINT minuto,hora

con lo que en la pantalla se debe presentar:

0 1

lo que refleja el hecho de que sus 59 minutos originales se han incrementado a 60 y el valor de la hora se ha incrementado en consecuencia.

Módulo 1.1.6: Trazado de las manillas

Habiendo calculado todos los números necesarios para llegar a la posición de las manillas podemos proceder ahora a su trazado.

Módulo 1.1.6: Líneas 6000 - 6110

```
6000 REM*******************************
6010 REM Trazado de las manillas
6020 REM*******************************
6030 longitud2=longitud*2/3
6040 PLOT 0,0,c
```

```
6050 DRAW longitud2*SIN(angulo-8),longit
ud2*COS(angulo-8)
6060 DRAW longitud*SIN(angulo),longitud*
COS(angulo)
6070 DRAW longitud2*SIN(angulo+8),longit
ud2*COS(angulo+8)
6080 DRAW 0,0
6090 MOVE longitud2*SIN(angulo),longitud
2*COS(angulo)
6100 GOSUB 7000
6110 RETURN
```

Comentarios

Línea 6030: Las manillas tendrán la forma de un rombo ligeramente alargado. La variable LONGITUD2 representa la distancia a la cual la manilla comienza a estrecharse hacia el punto.

Línea 6040: El cursor de gráficos se mueve hasta el centro de la esfera del reloj y el color se ajusta al valor de C, permitiendo así que el módulo anterior sea el que determine el color de lo que haya de ser dibujado.

Líneas 6050-6080: Estas líneas dibujan sobre la pantalla cuatro líneas que definen el contorno de la manilla. La primera línea tiene una longitud igual a LONGITUD2 y se traza formando un ángulo de ocho grados (en sentido antihorario) respecto al ángulo correcto que correspondería al minuto o a la hora correspondiente. La línea siguiente se traza desde el final de esta primera hasta una posición que está en el ángulo correcto, y a LONGITUD pixels del centro. La tercera línea se traza hasta una posición que está a LONGITUD2 pixels del centro y a ocho grados (en sentido horario) del ángulo correcto. Por último, el contorno se completa trazando una línea hasta el centro.

Líneas 6090-6100: Estas dos líneas son necesarias para el módulo siguiente, módulo que rellenará de color los contornos vacíos trazados por este módulo. La línea operativa es la 6090, que mueve el cursor de gráficos a un punto que está claramente dentro del contorno que se ha trazado.

Comprobación

Primero hemos de añadir una línea provisional para enlazar con el módulo siguiente:

```
7000 RETURN
```

Ahora teclee:

```
CLEAR:CLS:DEG:GOTO 5000[ENTER]
```

con lo que la pantalla debería limpiarse, y deberían dibujarse las dos manillas en una posición que indicara aproximadamente las doce y un minuto. Solamente estará dibujado el contorno de las manillas, ya que el proceso de colorearlas es llevado a cabo por el módulo siguiente.

Módulo 1.1.7: Rellenando una silueta con color

Una de las pocas deficiencias del 464, cuando se le compara con otros ordenadores personales, es la falta de una orden para rellenar de color una figura delimitada por su contorno. El módulo que damos aquí es bastante más lento de lo que sería una orden incorporada al sistema, y además tiene algunas limitaciones; en todo caso hace lo que se le pide, así que merece la pena tenerlo.

Tal y como está listada la rutina en este módulo rellenará cualquier polígono cuyos ángulos sean cóncavos cuando se les mira desde el exterior; es decir, cuando los ángulos apunten hacia afuera y no hacia adentro. Rellenará también *algunas* figuras que tengan ángulos apuntando hacia adentro, pero no todas. Si desea utilizar esta rutina sobre tales figuras irregulares, entonces tendrá que ir rellenándolas por secciones. Observe, en especial, que si la rutina se utiliza sobre una figura no cerrada probablemente quedará bloqueada en cuanto busque fuera de pantalla los contornos de la figura.

Módulo 1.1.7: Líneas 7000 - 7300

```
7000 REM*******************************
7010 REM Coloreado de las manillas
7020 REM*******************************
7030 IF TESTR(0,0)=c THEN RETURN
7040 s=2
7050 MOVE 2*INT(XPOS/2),2*INT(YPOS/2)
7060 :
7070 :
7080 REM buscar hacia arriba/derecha***
7090 IF TESTR(0,s)<>c THEN GOTO 7090
7100 IF TESTR(s,-s)<>c THEN GOTO 7090
7110 :
7120 :
7130 REM buscar hacia arriba/izquierda*
7140 MOVER -s,0
7150 IF TESTR(0,s)<>c THEN GOTO 7090
7160 IF TESTR(-s,-s)<>c THEN GOTO 7150
7170 :
7180 :
```

```
7190 REM colorear segun lineas horizonta
les**********************************
7200 x1=XPOS
7210 MOVER s,0
7220 PLOTR 0,0,c
7230 IF TESTR(s,0)<>c THEN GOTO 7220
7240 x2=XPOS-s
7250 MOVE x1,YPOS-s
7260 IF TESTR(s,0)<>c THEN GOTO 7290
7270 IF XPOS=x2 THEN RETURN
7280 GOTO 7260
7290 IF TESTR(-s,0)=c THEN GOTO 7200
7300 GOTO 7290
```

Comentarios

Línea 7030: La rutina, al igual que la mayoría de las rutinas de «pintado» o de «relleno», no funcionará si el color del pixel en el que se le ha dicho que comience tiene ya el color que se le ha dicho que emplee para el relleno.

Línea 7040: La variable S marca el paso mínimo horizontal que será tomado durante el curso del programa. Su valor se puede poner a cero si se están utilizando otros modos gráficos.

Línea 7050: En el modo 1 de gráficos, que es el que hemos estado utilizando para este programa, los pixels están agrupados por parejas. Esta línea asegura que, suceda lo que suceda con las coordenadas de trazado, ahora están ajustadas a números pares.

Líneas 7080-7100: Ahora comienza la búsqueda de alguna parte del contorno de la figura. Al principio estas líneas pueden parecer un tanto extrañas, ya que parece no haber ningún movimiento, tan sólo una línea que se bloquea a sí misma en un bucle infinito. En realidad, tanto TEST como TESTR (TEST Relativo) mueven el cursor de gráficos a la posición especificada; por lo tanto, la primera línea busca hacia arriba, en línea recta, el primer pixel del color correcto. Una vez que se ha encontrado una parte del contorno, la línea siguiente da un paso hacia abajo y a la derecha para ver si está libre. Si *está* libre entonces se llama de nuevo a la primera línea para ver si es posible ir más allá.

Líneas 7130-7160: Habiendo buscado hacia arriba y hacia la derecha en toda su extensión, estas líneas comienzan a buscar hacia la izquierda tratando de encontrar un camino que conduzca más hacia arriba todavía. Al final de estas líneas y de las del grupo anterior, el programa ha encontrado la parte superior de la figura o bien ha llegado a un callejón sin salida, si es que la figura no es regular.

Líneas 7200-7230: Habiendo encontrado lo que debe de ser el punto más alto, el proceso de «relleno» puede comenzar en serio. Las órdenes MOVER (MOVE Relativo) y PLOTR (PLOT Relativo) son utili-

zadas para trazar una línea de pixels de izquierda a derecha hasta que se alcanza otra parte del contorno.
Líneas 7240-7250: Habiendo finalizado una línea horizontal, la variable X2 se hace igual a la posición más hacia la derecha de las que se han rellenado, y se hace retroceder el cursor hasta el comienzo de la línea pero un pixel más abajo.
Líneas 7260-7280: Esta comprobación se realiza cuando se hace retroceder el cursor hacia la izquierda. Si el pixel al que se llega ya está coloreado con el color de relleno, el programa explora de izquierda a derecha en busca de un pixel en blanco. Si la búsqueda continúa más allá de la línea superior (la que acaba de ser rellenada), entonces es que se ha alcanzado el fondo de la figura.
Leas 7290-7300: Si al mover el cursor hacia abajo y hacia la izquierda, para comenzar una nueva línea, se llega a un pixel que está en blanco, el programa busca hacia la izquierda hasta encontrar el borde de la figura, comenzando entonces a rellenar la línea correspondiente.

Comprobación

La comprobación más sencilla para este módulo es seguir el mismo procedimiento de comprobación del módulo anterior. La única diferencia en el resultado debería ser que las dos manillas ahora están pintadas con el color correcto.

Módulo 1.1.8: Haciendo que todo funcione conjuntamente

A estas alturas es muy posible que se esté preguntando por qué se han escrito los programas en la forma en que se ha hecho. Es que no se podían haber puesto juntas todas las funciones que se han descrito y haberlas lanzado con el uso de unos cuantos GOTO. Desgraciadamente, esa es la forma en que están construidos y publicados muchos programas.

En este libro encontrará que todos los programas están construidos a partir de módulos fácilmente identificables. La razón es que los programas escritos de esta forma pueden leerse con más facilidad, pueden depurarse mejor, pueden cambiarse sustituyendo módulos por otros que funcionan con más eficacia (cuando aprenda nuevos métodos), y pueden ampliarse parcheando más módulos. Hay mucho que aprender de los programas de este libro, pero probablemente la lección más valiosa de todas (para su programación futura) será la técnica de programación modular.

Este módulo es la clave de dicha técnica. Cuando se han introducido y comprobado todos los módulos de trabajo necesitamos uno más para que controle el flujo del programa. En cierto sentido, el mó-

dulo que está a punto de meter *es* el programa en sí, todo lo demás es simplemente una extensión de él.

Módulo 1.1.8: Líneas 1000 - 1100

```
1000 REM********************************
1010 REM Control
1020 REM********************************
1030 GOSUB 2000
1040 GOSUB 3000
1050 GOSUB 4000
1060 GOSUB 5000
1070 EVERY 3000 GOSUB 5000
1080 IF INKEY$<>" " THEN 1080
1090 CLS
1100 END
```

Comentarios

Línea 1070: La clave del programa. Se trata de la línea que proporciona la sincronización del programa utilizando para ello la potente orden EVERY del 464. La función de esta orden es permitir al usuario la especificación de un intervalo de tiempo de forma que, cada vez que transcurra ese período, se interrumpa cualquier tarea que se esté ejecutando y se realice otra. Cuando finaliza la interrupción, la tarea original recomienza, sea cual fuere, hasta la siguiente interrupción. El intervalo tras el cual ocurrirá la interrupción se mide en cincuentavos (1/50) de segundo, por lo que esta línea llama al módulo que comienza en la línea 5000 una vez cada minuto, incrementando por lo tanto el valor de los minutos y retrazando las manillas.

Líneas 1080-1100: El resto del tiempo el programa estará bloqueado por el bucle infinito representado por la línea 1080, en espera de que el usuario pulse la barra espaciadora. En cuanto esto ocurre, se borra la pantalla y el programa finaliza.

Comprobación

Ahora debería funcionar todo el programa. Córralo, introduzca la hora y deberá presentarse la esfera del reloj con las manillas dibujadas en la posición correcta. Se puede hacer una comprobación más rápida del funcionamiento de la presentación visual, sin más que suprimir dos ceros del valor 3000 de la línea 1070. Esto dará lugar a que las manillas sean trazadas, borradas y retrazadas constantemente.

PROGRAMA 1.2: Reloj

Función del programa

Una de las cosas más divertidas de los ordenadores que tienen presentaciones gráficas tan buenas como las del CPC464 es que le permiten jugar a presentar las cosas en formas nuevas e imaginativas. Recién metido un reloj, bastante estándar, este otro programa proporciona una presentación bastante diferente de la hora. En Reloj, las horas y los minutos están representados por dos líneas que recorren la pantalla de izquierda a derecha y de arriba a abajo, dividiéndola en cuatro rectángulos de colores diferentes. Gran parte del material de Reloj es similar al de Analoj, de forma que las explicaciones pueden ser acortadas consecuentemente.

Los temas nuevos que introduce el programa son:

1) Ventanas.
2) Formateado de números.
3) Cortado en rebanadas o disección de una cadena.

Módulo 1.2.1: Inicialización

Se trata de un módulo sencillo de inicialización para ajustar los colores de la presentación. El propósito de la ventana definida en la línea 2100 se explicará en los comentarios de un módulo posterior.

Módulo 1.2.1: Líneas 2000 - 2120

```
2000 REM********************************
2010 REM Inicializacion
2020 REM********************************
2030 MODE 1
2040 INK 0,1
2050 INK 1,24
2060 INK 2,9
2070 INK 3,6
2080 BORDER 1
2090 PAPER 0:PEN 1
2100 WINDOW #1,39,39,9,18
2110 PAPER #1,0:PEN #1,1
2120 RETURN
```

Comprobación

Igual que en el primer programa, la única comprobación que puede hacerse a estas alturas es teclear:

GOTO 2000[ENTER]

lo que debería resultar en la finalización del programa con un mensaje de error «Unexpected RETURN». Cualquier otro mensaje de error indicará que algo se ha hecho mal al introducir el módulo.

Módulo 1.2.2: Introducción de la hora

El mismo módulo que en Analoj:

Módulo 1.2.2: Líneas 3000 - 3090

```
3000 REM*******************************
3010 REM Ajuste de la hora
3020 REM*******************************
3030 PRINT "Ajuste la hora:":PRINT
3040 INPUT "Hora (1 a 12):";hora
3050 IF hora<1 OR hora>12 THEN PRINT "**
*Fuera del intervalo: introduzca dato ot
ra vez***":GOTO 3040
3060 INPUT "Minutos (0 a 59):";minuto
3070 IF minuto<0 OR minuto>59 THEN PRINT
 "***Fuera del intervalo: introduzca dat
o otra vez***":GOTO 3060
3080 segundo=0
3090 RETURN
```

Módulo 1.2.3: Preparación del borde de la pantalla

Este módulo dibuja las marcas correspondientes a las horas y a los minutos, sobre el borde izquierdo y el superior de la pantalla.

Módulo 1.2.3: Líneas 4000 - 4120

```
4000 REM*******************************
4010 REM Preparacion de la presentacion
4020 REM*******************************
4030 CLS:PRINT"  ";
4040 FOR i=5 TO 55 STEP 10
4050 PAPER 0:PEN 1:PRINT USING"###";i;
```

```
4060 PAPER 1:PEN 0:PRINT USING"###";i+5;
4070 NEXT i
4080 PAPER 0:PEN 1
4090 FOR i=1 TO 12
4100 LOCATE 1,i*2+1:PRINT USING"##";i
4110 NEXT i
4120 RETURN
```

Comentarios
Líneas 4040-4070: El bucle presenta los valores de los minutos espaciados a lo largo del borde superior de la pantalla, alternando rótulos normales e inversos (permutando los colores del papel y de la pluma). Se utiliza PRINT USING para asegurar que cada cifra (múltiplo de cinco minutos) ocupa el espacio correspondiente a tres caracteres; puesto que cualquiera de los números presentados tiene menos de tres caracteres, PRINT USING los rellenará por la izquierda con uno o dos espacios en blanco.
Líneas 4080-4110: La variable I del bucle se utiliza, en unión con la orden LOCATE, para presentar las horas en el borde izquierdo de la pantalla y en sentido descendente.

Comprobación
Para comprobar el módulo de forma eficaz necesitará intercalar una línea provisional:

```
4115 GOTO 4115
```

Su objetivo es impedir el desplazamiento ascendente (enrollamiento) de la pantalla por la aparición del mensaje de error «Unexpected RETURN». Ahora teclee:

```
GOTO 4000[ENTER]
```

con lo que debería ver la escala de las horas y los minutos impresa junto al borde de la pantalla. Cuando esté satisfecho pulse dos veces la tecla ESCAPE y no olvide suprimir la línea provisional.

Módulo 1.2.4: Creación de una cadena horaria

Una de las funciones del programa será presentar la hora en forma digital, así como por el método más novedoso descrito anteriormente. La cadena está basada en variables que serán creadas por el módulo siguiente, aunque éste debe introducirse primero para permitir que aquel se ejecute correctamente.

Módulo 1.2.4: Líneas 7000 - 7050

```
7000 REM*******************************
7010 REM Creacion de una cadena horaria
7020 REM*******************************
7030 hor$=STR$(1000000+hora*10000+minuto
*100+segundo)
7040 hor$=MID$(hor$,3,2)+"  "+MID$(hor$,
5,2)+"  "+MID$(hor$,7,2)
7050 RETURN
```

Comentarios

Líneas 7030-7040: Un sistema sencillo para combinar una serie de números en un número es multiplicar cada número por potencias decrecientes de 10 y sumar entonces todo junto a una potencia más alta de 10. Si, por ejemplo, HORA=1, MINUTO=36 y SEGUNDO=2 el resultado de esta línea sería 1013602. ¿Por qué? Bien, eche un vistazo a la cifra de las horas, el 1000000 que se sumó ha colocado un 0 delante del único «1» del valor de la hora, como lo ha hecho con el «2» de los segundos.

Ahora utilizamos lo que se conoce como «cortado de una cadena en rebanadas» o «disección de una cadena» para separar tres secciones de dos dígitos a partir del número (seguros ya de que se ha añadido un cero a la izquierda en caso de que alguno de los valores originales constara de un único dígito). Las funciones de cadena de la segunda línea extraen las tres cifras, especificando la posición de comienzo en el número creado y el número de caracteres a extraer. Así, por ejemplo, MID$(A$,3,2) significa la parte de A$ que comienza en el carácter 3° y que tiene una longitud de dos caracteres.

Si mira las órdenes de disección de la línea 7040, puede que quede confundido por el hecho de que el valor de la hora se toma desde el carácter 3 en adelante, cuando comienza indudablemente en el dígito 2 de la cadena artificial creada en la línea 7030. La respuesta a esta contradicción aparente se encuentra en el hecho de que, para extraer los valores a partir de una cadena, primero hemos tenido que transformar el número en esa cadena usando la función STR$. Cuando hacemos esto, el número aparenta exactamente lo mismo, aunque ahora puede ser tratado como una cadena; la única diferencia es que, si el número es positivo, se añade un espacio a la izquierda para suplir la falta de un signo «+». El primer carácter utilizable de un número que ha sido transformado en cadena mediante STR$ es el número 2 de carácter.

El resultado de esta manipulación es que terminamos con una cadena de la forma:

HORA EN DOS DIGITOS/ESPACIO/MINUTOS EN DOS DIGITOS/ESPACIO/ SEGUNDOS EN DOS DIGITOS

Comprobación

Teclee:

HORA=1[ENTER]
MINUTO=1[ENTER]
SEGUNDO=1[ENTER]
GOTO 7000[ENTER]

y cuando el programa retorne con un mensaje de error «Unexpected RETURN», teclee:

PRINT hor$[ENTER]

con lo que debería ver:

01 01 01

Módulo 1.2.5: Ajustes horarios

Este módulo es comparable al módulo de ajustes horarios del programa anterior.

Módulo 1.2.5: Líneas 6000 - 6160

```
6000 REM*******************************
6010 REM Ajustes horarios
6020 REM*******************************
6030 segundo=segundo+1
6040 WHILE segundo>59
6050 segundo=0:minuto=minuto+1
6060 WHILE minuto>59
6070 minuto=0:hora=hora+1
6080 WHILE hora>12
6090 hora=1
6100 WEND
6110 WEND
6120 WEND
6130 LOCATE #1,1,1
6140 GOSUB 7000
6150 PRINT#1,hor$;
6160 RETURN
```

Comentarios
Líneas 6030-6120: Estos tres bucles anidados escalonan cuidadosamente la serie de operaciones que pueden ser necesarias cuando se incrementa en 1 el valor de los segundos. Cada uno de los bucles se activa solamente si, tras el cambio de los segundos, ha de ajustarse el valor de la hora o de los minutos; es decir, cuando se sobrepasan los valores tope de las horas o de los minutos.
Líneas 6130-6160: Habiendo creado en el módulo anterior una cadena que representa la hora, estas líneas la presentan en formato vertical. La utilización de la ventana que se definió en el módulo de inicialización hace que esto sea una cuestión sencilla. Si retrocede hasta dicho módulo, observará que la ventana se definió a la derecha de la pantalla y con una *anchura de un sólo carácter*, por lo que cualquier cosa que salga por la ventana aparecerá con cada carácter bajo el anterior.

Comprobación
Teclee:

```
CLEAR
SEGUNDO=59
MINUTO=59
HORA=12
GOTO 6000[ENTER]
```

con lo que debe aparecer el mensaje «Unexpected RETURN» y un «01 00 00» presentado verticalmente y a la derecha de la pantalla, demostrando que los bucles de la primera mitad del programa son capaces de actualizar correctamente los tres valores.

Módulo 1.2.6: Presentación de la hora

En este módulo abordaremos la tarea de visualizar en pantalla los rectángulos que representarán la hora. Lo que el módulo pretende conseguir es el efecto de una línea, representando los minutos, que barre la pantalla en sentido horizontal, y de otra descendente que registra las horas. Esto se consigue dividiendo la pantalla en cuatro secciones rectangulares, dos rojas y dos verdes, cuyos bordes representan las líneas de las horas y de los minutos (fig. 1.2).

Sin duda que sería posible colocar directamente estos rectángulos en pantalla y en la posición correcta, pero es una dificultad innecesaria. Todo lo que realmente necesitamos hacer es trazar, en la mitad superior de la pantalla, líneas con espacios coloreados que

cambian de rojo a verde en el punto apropiado, y en la mitad inferior de la pantalla líneas que cambian de verde a rojo. Las líneas en sí serán creadas mediante un bucle sencillo y la orden LOCATE.

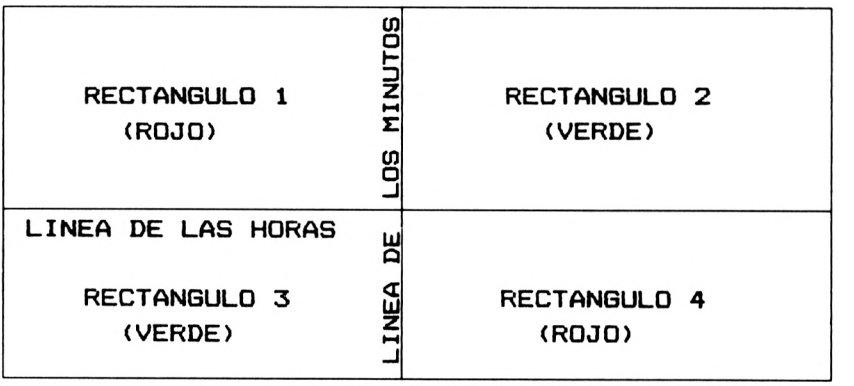


Fig. 1.2 La pantalla dividida en cuatro secciones rectangulares.

Módulo 1.2.6: Líneas 5000 - 5100

```
5000 REM*********************************
5010 REM Presentacion de la hora
5020 REM*********************************
5030 PAPER 2:PEN 3
5040 FOR i=2 TO 25
5050 IF i=INT(hora*2+minuto/30+2) THEN P
APER 3:PEN 2
5060 LOCATE 3,i
5070 PRINT STRING$(minuto*0.6,CHR$(143))
;TAB (39)
5080 NEXT i
5090 RETURN
5100 RETURN
```

Comentarios

Línea 5030: Se fija el color del papel a verde y el de impresión a rojo.

Líneas 5040-5080: Se presentan las 24 líneas de visualización (representando cada línea media hora). La visualización comienza en la línea 2 de la pantalla (numerando a partir de 0).

Línea 5050: Si la variable de bucle ha alcanzado la posición donde debería estar la línea de las horas, entonces se permutan los colores de papel e impresión.

Líneas 5060-5070: Estas dos líneas sitúan e imprimen una de las líneas de la presentación. La función STRING$ se utiliza para crear

una línea de espacios inversos (CHR$(143)) hasta la línea de división de los minutos. El TAB del final de la línea mueve la posición de impresión al final de la línea de pantalla, poniendo cualesquiera espacios sin utilizar con el color actual del papel (PAPER). La longitud de la cadena creada estará basada en el valor de la variable MINUTO, pero será ajustada (multiplicada por 0,6) para tener en cuenta el hecho de que la presentación tiene una anchura de 36 caracteres, no de 60.

Comprobación

Teclee, en modo directo:

HORA=6[ENTER]
MINUTO=30[ENTER]
GOTO 5000

La pantalla debería dividirse en cuatro rectángulos aproximadamente iguales, con bordes coincidentes con las líneas de la ilustración anterior. El programa debe entonces detenerse con un mensaje de error «Unexpected RETURN».

Módulo 1.2.7: Juntando todo

Habiendo introducido todos los elementos de trabajo del programa, podemos construir ahora un módulo de control que los ejecute en el orden correcto:

Módulo 1.2.7: Líneas 1000 - 1090

```
1000 REM********************************
1010 REM Control
1020 REM********************************
1030 ON BREAK GOSUB 1090
1040 GOSUB 2000
1050 GOSUB 3000
1060 EVERY 50 GOSUB 6000
1070 GOSUB 4000
1080 GOSUB 5000:GOTO 1080
1090 PEN 1:PAPER 0:CLS:CLEAR:END
```

Comentarios

Línea 1030: La orden ON BREAK GOSUB nos permite definir lo que hará el programa si se pulsa dos veces la tecla ESCAPE. En este caso deseamos que se salte al final del programa, donde se inicializan los colores, se borra la memoria y se acaba dicho programa.

Línea 1060: Una vez más la orden EVERY suministra la clave para la sincronización del programa. Esta llamada al módulo de ajustes horarios se efectúa cada segundo.

Comprobación

Ahora está en posición de ejecutar todo el programa, introduzca la hora y véala representada.

PROGRAMA 1.3: Temporizador

Función del programa

El programa le proporciona 16 temporizadores flexibles de cuenta atrás, cada uno de los cuales puede ser programado independientemente para que haga sonar una señal de alarma (tras un período de tiempo especificado) y mostrar un breve mensaje que indique el suceso en particular por el que se ha producido el aviso.

Cuando examine el programa observará que su diseño es diferente del de los que le han precedido. Puesto que es el programa más largo y más complejo de los que hasta ahora se ha encontrado en el libro se han insertado líneas de programa en blanco (en esta ocasión) para ayudarle a identificar las unidades de trabajo que componen el programa. Estas líneas en blanco no son necesarias para el funcionamiento del programa y puede omitirlas si así lo desea.[1] El programa es uno de los pocos que va a ser diseñado de esta forma, por la sencilla razón de que el espacio extra necesario para hacer la misma operación con cada uno de los programas hubiera exigido la disminución del material total del libro.

```
Ahora son las:   06:30:10

          TEMPORIZADORES
          ==============

1) 07:00:00 Despierta
2) 07:15:00 Digo que te despiertes
3) 07:30:00 Es tu ultima oportunidad
4) 08:00:00 Has perdido el tren
5) 17:45:00 Los pitufos en la TV
6) 20:30:00 Llamar a la abuela
```

Fig. 1.3 Visualización típica del programa Temporizador.

1. En la versión castellana se han omitido. (*N. del T.*)

Las nuevas técnicas abarcadas en este programa son:

1) El módulo «menú».
2) El uso sencillo de SOUND.
3) Insertar y borrar elementos en matrices.
4) La creación de un estado de espera.

Módulo 1.3.1: Inicialización

Un módulo más complejo que los empleados en los programas anteriores, lo que no hace más que reflejar la mayor complejidad del programa en conjunto. La mayor parte del módulo está dedicada a la tarea de preparar la pantalla (incluyendo dos ventanas) a utilizar por el resto del programa.

Módulo 1.3.1: Líneas 2000 - 2150

```
2000 REM********************************
2010 REM Inicializacion
2020 REM********************************
2030 MODE 1
2040 GOSUB 12000
2050 WINDOW 1,40,25,25
2060 PAPER 0:PEN 1:CLS
2070 WINDOW #1,3,40,1,1
2080 PAPER #1,0:PEN #1,1:CLS #1
2090 PRINT #1,"Ahora son las:"
2100 WINDOW #2,1,40,3,23
2110 PAPER #2,3:PEN #2,2:CLS #2:PRINT #2
2120 SPEED INK 50,25
2130 DIM thora(15),tminuto(15),ttemp(15)
,mensaje$(15)
2140 ttemp(0)=-1
2150 RETURN
```

Comentarios
Línea 2040: En contraste con los programas anteriores, este módulo no tendrá ninguna instrucción INK. Esto es así porque, en ocasiones, se cambiarán los colores de tinta (INK) durante el programa. El GOSUB de esta línea llama a la pequeña rutina que ajusta los colores de comienzo; esta subrutina se introducirá inmediatamente después de la actual.
Línea 2050: La orden WINDOW se puede utilizar en dos formas, bien para reservar un área de pantalla a la que se puede llamar utilizando

un «número de cauce o canal» (método que se utilizó en el programa Reloj), o bien especificando simplemente el tamaño de la ventana. La segunda de las formas, la empleada en esta línea, ajusta la ventana implícita o de omisión; es decir, aquella en la que (a menos que se especifique lo contrario) se visualiza normalmente la impresión. En este caso la ventana implícita se fija a una única línea situada en la parte inferior de la pantalla.
Líneas 2070-2120: Estas líneas preparan dos ventanas más. Una de ellas (la #1) es simplemente una única línea en la parte superior de la pantalla, mientras que la otra (la #2) forma el cuerpo principal de dicha pantalla. La orden SPEED INK se utiliza cuando un número de tinta tiene dos colores asignados, y especifica la velocidad de parpadeo, entre los dos colores, de las letras impresas con ese número de tinta.
Línea 2130: Las matrices sobre las que trabajará el programa. Sus usos particulares se describirán a lo largo de los comentarios de la parte principal del programa.

Módulo 1.3.2: Colores de pantalla al comenzar

Como se explicó en los comentarios del último módulo, los colores de pantalla pueden cambiar en el transcurso del programa. Este pequeño módulo fija los colores de comienzo.

Módulo 1.3.2: Líneas 12000 - 12080

```
12000 REM******************************
12010 REM Encendido de la pantalla
12020 REM******************************
12030 INK 0,1
12040 INK 1,24
12050 INK 2,26
12060 INK 3,2
12070 BORDER 1
12080 RETURN
```

Módulo 1.3.3: Ajuste inicial de la hora

El módulo estándar que se encuentra en Analoj, con la excepción de que la hora se introduce en formato de 24 horas.

Módulo 1.3.3: Líneas 3000 - 3100.

```
3000 REM*******************************
3010 REM Ajuste de la hora
3020 REM*******************************
3030 CLS:PRINT "Ajuste la hora:":PRINT
3040 INPUT "Hora (1 a 12):";hora
3050 IF hora<1 OR hora>12 THEN PRINT "**
*Fuera del intervalo: introduzca dato ot
ra vez***":GOTO 3040
3060 INPUT "Minutos (0 a 59):";minuto
3070 IF minuto <0 OR minuto>59 THEN PRIN
T "***Fuera del intervalo: introduzca da
to otra vez***":GOTO 3060
3080 segundo=0
3090 hor=hora*60+minuto
3100 RETURN
```

Módulo 1.3.4: Ajustes horarios

Un módulo similar a los módulos de ajustes horarios anteriores.

Módulo 1.3.4: Líneas 14000 -14180.

```
14000 REM*******************************
14010 REM Ajustes horarios
14020 REM*******************************
14030 segundo=segundo+5
14040 WHILE segundo>59
14050 segundo=0:minuto=minuto+1
14060 WHILE minuto>59
14070 minuto=0:hora=hora+1
14080 WHILE hora>23
14090 hora=0
14100 WEND
14110 WEND
14120 WEND
14130 LOCATE#1,20,1
14140 h=hora:m=minuto:s=segundo:GOSUB 15
000
14150 PRINT#1,hor$;
14160 hor=hora*60+minuto
14170 GOSUB 9000
14180 RETURN
```

Módulo 1.3.5: Formateado de la hora

El módulo de ajustes horarios, como en el programa anterior, funciona conjuntamente con este otro que transforma la hora actual en una forma entendible por el usuario.

Módulo 1.3.5: Líneas 15000 - 15050

```
15000 REM*******************************
15010 REM Creacion de una cadena hora-
          ria
15020 REM*******************************
15030 hor$=STR$(1000000+h*10000+m*100+s)
15040 hor$=MID$(hor$,3,2)+":"+MID$(hor$,
5,2)+":"+MID$(hor$,7,2)
15050 RETURN
```

Módulo 1.3.6: Comprobación de los temporizadores y activación alarma

Este módulo es una parte integral del programa (es necesaria pero no es su esencia), en el sentido de que permite a otras partes del programa introducir un estado de espera durante el cual el usuario puede hacer una entrada, pero al mismo tiempo el programa continúa constantemente el trabajo de muestrear los temporizadores, para ver si alguno de ellos ha vencido y, en caso necesario, hacer sonar la señal de alarma. El módulo lo introduciremos ahora, aun cuando no será utilizado por completo hasta que se hayan introducido varios módulos más, pues es una subrutina esencial para el módulo de menú que viene a continuación.

Módulo 1.3.6: Líneas 9000 - 9300

```
9000 REM*******************************
9010 REM Inspeccion temporizador
9020 REM*******************************
9030 IF ttemp(0)<>hor THEN RETURN
9040 CLS #2: PRINT #2
9050 GOSUB 12000:INK 2,26,2
9060 PRINT #2,TAB(19);"AVISO";TAB(19);"=
====":LOCATE #2,1,11
9070 longm=LEN(mensaje$(0))
9080 PRINT #2,TAB(19-longm/2);STRING$(lo
ngm+4,"*")
9090 PRINT #2,TAB(19-longm/2);"*";TAB(22
+longm/2);"*"
```

```
9100 PRINT #2,TAB(19-longm/2);"* ";mensa
je$(0);" *"
9110 PRINT #2,TAB(19-longm/2);"*";TAB(22
+longm/2);"*"
9120 PRINT #2,TAB(19-longm/2);STRING$(lo
ngm+4,"*")
9130 nota=250
9140 WHILE INKEY$="" AND ttemp(0)=hor
9150 WHILE (SQ(1) AND 7)>0
9160 SOUND 1,nota,15,5
9170 nota=750-nota
9180 WEND
9190 WEND
9200 sonando=1
9210 FOR i=1 TO temporiz-1
9220 thora(i-1)=thora(i)
9230 tminuto(i-1)=tminuto(i)
9240 ttemp(i-1)=ttemp(i)
9250 mensaje$(i-1)=mensaje$(i)
9260 NEXT i
9270 temporiz=temporiz-1
9280 IF temporiz=0 THEN ttemp(0)=-1
9290 INK 2,26
9300 RETURN
```

Comentarios

Línea 9030: Nuestra primera referencia de trabajo a la matriz TTEMP que se utiliza para almacenar las horas de vencimiento de los temporizadores. Otros módulos posteriores, que permitirán al usuario ajustar los temporizadores, se encargarán de que el temporizador que tenga la hora más temprana esté colocado siempre en la primera posición de la matriz (o sea su elemento 0). En la inmensa mayoría de las veces que se llame a esta subrutina, su ejecución finalizará en esta línea; solamente si la hora actual (HOR, que fue calculada por el módulo de ajustes horarios) es la misma que la hora de alarma almacenada en el primer elemento de TTEMP, sonará la alarma.

Líneas 9040-9120: Estas líneas parecen bastante más complejas de lo que son en realidad. Su propósito es presentar la palabra «AVISO» por la ventana #2 (la parte principal de la pantalla), junto con cualquier mensaje asociado al temporizador activado. El mensaje estará ribeteado de asteriscos y estará centrado en pantalla, cosa que se consigue poniendo el primer carácter del mensaje en la posición determinada por la fórmula:

POSICION MEDIA DE LA PANTALLA-LONGITUD DE LA CADENA A IMPRIMIR/2

Esta fórmula la puede ver empleada en las líneas 9090-9120, donde toma la forma «19-longm/2».
Líneas 9130-9200: Estas líneas hacen sonar (mediante un SOUND) una señal de aviso de dos tonos durante un minuto (mientras la hora de vencimiento del temporizador es la misma que la hora actual en minutos). La señal cesa de sonar al pulsar una tecla. De los dos bucles, el interior es el que hace sonar la nota propiamente dicha, siendo la línea 9150 la que asegura que cada nota es retenida hasta que la función SQ indica que la nota anterior ha dejado de sonar. La variable SONANDO se pone a 1 para indicar al módulo de menú que se ha hecho sonar una alarma; para ver la razón que hay tras esto vea los comentarios del menú.
Líneas 9210-9280: Una vez que ha sonado una alarma, estas líneas se encargan de borrarla de las matrices que almacenan las horas y los mensajes de cada temporizador. Esto se consigue desplazando todos los datos, del segundo en adelante, a una posición anterior dentro de la matriz, eliminando así al primer elemento que justamente es el que acaba de provocar la señal de alarma. La variable TEMPORIZ registra cuántos temporizadores ajustados hay en cada momento, por lo que consiguientemente debe ser reducida en 1. Por último, si no queda ningún temporizador ajustado, la línea 9280 se asegura de colocar una hora inexistente en la posición del primer temporizador, de forma que no pueda sonar involuntariamente a ninguna hora.

Módulo 1.3.7: El menú del programa

Llegamos ahora a una nueva técnica que desempeñará un papel importante en los programas de este libro: el «menú». En los programas que nos han preparado el terreno para éste, el control se dejó en manos de los propios programas. Una vez lanzados (con un RUN), un módulo de control tomaba el mando y dictaba el flujo de ejecución del programa hasta que el usuario indicaba que el programa debía finalizar.

Este programa (y muchos de los que siguen) es diferente en el sentido de que no hay una única dirección para el flujo del programa. El programa presenta al usuario una gama de posibilidades y debe ser éste quien, en gran parte, decida lo que debe suceder. Esto se hace por medio de un módulo al que se le conoce como menú del programa, módulo que presenta al usuario una lista de opciones que proporciona el programa y que le permite especificar cuál de ellas elige. Programas posteriores y más complejos harán uso de varios menús, donde cada uno de ellos refleja una gama de opciones con un determinado encabezamiento; por el momento, sin embargo, nos atendremos al único menú necesitado por este programa.

```
4000 REM*********************************
4010 REM Menu
4020 REM*********************************
4030 WHILE x$<>"5"
4040 CLS #2:PRINT #2
4050 PRINT #2,TAB(19);"MENU";TAB(19);"==
==":PRINT #2
4060 PRINT #2," 1) Ajustar temporizador.
":PRINT #2
4070 PRINT #2," 2) Anular temporizador."
:PRINT #2
4080 PRINT #2," 3) Mostrar los temporiza
dores y esperar     hasta que se pulse un
a tecla.":PRINT #2
4090 PRINT #2," 4) Borrar la pantalla y
esperar hasta       que se pulse una tecl
a.":PRINT #2
4100 PRINT #2," 5) Finalizar programa.":
PRINT #2
4105 PRINT #2,"****PULSAR NUMERO CORRESP
ONDIENTE A LA       OPCION DESEADA****"
4110 sonando=0:x$=""
4120 WHILE x$="" AND sonando=0
4130 x$=INKEY$
4140 IF sonando THEN x$=""
4150 WEND
4160 ON VAL(x$) GOSUB 5000,6000,7000,800
0
4170 PRINT #2
4180 WEND
4190 RETURN
```

Comentarios

Líneas 4030-4170: Este bucle continuará visualizando el menú, hasta que el usuario introduzca el número 5, cada vez que la ejecución del programa retorne al módulo. Cuando el usuario pulsa el 5, la ejecución del programa retorna al módulo de control, módulo que aún no hemos introducido.

Líneas 4040-4100: Estas líneas presentan la lista de las opciones del programa por la ventana #2, la ventana principal que ocupa la mayor parte de la pantalla.

Líneas 4110-4150: Generalmente, en este punto del módulo de menú debería haber una sentencia INPUT corriente. La razón de que se incluya este conjunto de líneas más complejo para llamar al módulo

anterior es que queremos muestrear continuamente el estado de los temporizadores, incluso con el menú en pantalla. Esto no se puede hacer si se utiliza INPUT, ya que un INPUT bloquea el programa hasta que se pulsa ENTER (ni siquiera un EVERY puede interrumpir a un INPUT).

El método empleado por las líneas es poner una cadena vacía en la variable (X$) que se utilizará para almacenar la entrada del usuario, y seguir ejecutando el bucle en tanto permanezca vacía. Si el usuario pulsa una tecla se detectará mediante el INKEY$ de la línea 4130, y el carácter asociado a la tecla se almacenará en X$, finalizando así el bucle.

No obstante, mientras transcurre este muestreo del teclado, el módulo anterior está siendo llamado constantemente para ver si es necesario hacer sonar una alarma. Dicho sea de paso, la variable SONANDO tiene un uso bastante claro. Normalmente no ocurrirá nada cuando se llama al módulo anterior, de forma que el menú puede seguir tranquilamente en pantalla; sin embargo, si se hace sonar una alarma, el menú será borrado de la pantalla para presentar el aviso y un mensaje (en este caso, SONANDO se pondrá a 1 para indicar al módulo de menú que tal menú debe volver a ser presentado en pantalla).

Línea 4160: La entrada hecha por el usuario es transformada en un número, si es que es posible, mediante la función VAL. La orden ON...GOSUB selecciona de entre la lista de destinos posibles el que corresponde al número introducido por el usuario. Si se introduce un número que está fuera del intervalo de la lista de destinos de GOSUB; por ejemplo, si el número es 0 o mayor que el número de destinos de la lista entonces la orden ON...GOSUB no actúa y el bucle muestra otra vez el menú.

Comprobación

Antes de comprobar el programa tal y como está ahora mismo, tiene más sentido introducir el breve módulo de control que viene a continuación.

Módulo 1.3.8: El módulo de control

Mucho más simple que en otros programas, porque la mayor parte del trabajo está siendo soportado por el menú. Una vez más, no obstante, la sincronización para el programa se realiza mediante un EVERY incorporado a este módulo. En este caso la sincronización se hace cada cinco segundos, como hicimos observar en los comentarios del módulo de ajustes horarios.

Módulo 1.3.8: Líneas 1000 - 1080

```
1000 REM*********************************
1010 REM Control
1020 REM*********************************
1030 GOSUB 3000
1040 GOSUB 2000
1050 EVERY 250 GOSUB 14000
1060 GOSUB 4000
1070 MODE 1
1080 END
```

Comprobación

El lanzamiento del programa debe dar lugar a la petición de introducción de la hora y de los minutos, seguido por el menú del programa. Introduzca algunos números incorrectos para confirmar que no son aceptados. Si se especifican las opciones de programa 1 a 4, el programa se detendrá con el mensaje de error: «Line does not exist» (No existe esa línea), mientras que la opción 5 borrará la pantalla y terminará el programa.

Módulo 1.3.9: Visualización del ajuste de los temporizadores activados

El propósito del módulo es sacar por pantalla, de forma ordenada, las horas a las que están ajustados los 16 temporizadores (lógicamente sólo aquellos que estén activados). Por el momento la pantalla quedaría en blanco, ya que aún no hemos introducido el módulo que nos permite ajustar los valores de dichos temporizadores; sin embargo, la introducción del módulo nos permitirá comprobar el de ajuste de los valores en cuanto lo introduzcamos.

Módulo 1.3.9: Líneas 7000 - 7170

```
7000 REM*********************************
7010 REM Mostrar temporizadores y espe-
     rar
7020 REM*********************************
7030 k$=""
7040 WHILE k$=""
7050 CLS #2:PRINT #2
7060 PRINT #2,TAB(12);"TEMPORIZADORES";T
AB(12);"==============":PRINT #2
7070 FOR i=0 TO temporiz-1
```

```
7080 h=thora(i):m=tminuto(i):s=0:GOSUB 1
5000
7090 PRINT #2,USING " ##) &";i+1;hor$+"
"+mensaje$(i)
7100 NEXT i
7110 sonando=0
7120 WHILE k$="" AND sonando=0
7130 k$=INKEY$
7140 IF sonando=1 THEN k$=""
7150 WEND
7160 WEND
7170 RETURN
```

Comentarios

Líneas 7030-7040 y 7160: La presentación permanecerá en pantalla hasta que se pulse una tecla.

Líneas 7070-7100: Como se hizo observar con anterioridad, la variable TEMPORIZ registra el número de temporizadores ajustados en cada momento. Este bucle presenta en pantalla los ajustes del número de temporizadores indicado por dicha variable, utilizando el Módulo 1.3.5 para crear una presentación en cadena de la hora según está registrada en las matrices THORA y TMINUTO, y utilizando PRINT USING para asegurar una lista ordenada (bien presentada). El especificador «&» que aparece en esta instrucción puede confundir a algunos, su propósito es permitirnos mencionar un número y después una cadena en la misma instrucción. Si no fuera por el «&» se hubiera generado un error «Type mismatch» (Tipos no coincidentes) cuando el programa intentara interpretar HOR$ como un número en la línea 7090.

Líneas 7110-7150: Como en el menú se trata de un estado de espera durante el cual una orden EVERY estará muestreando los temporizadores. Observe que si se activa una alarma mientras el bucle está en funcionamiento, y el usuario pulsa una tecla para desactivarla, la presentación de los temporizadores quedará sin concluir. La línea 7140 asegura que, si la variable SONANDO indica que ha estado sonando una alarma, se tiene que pulsar una tecla otra vez para que el programa vuelva al menú.

Comprobación

Teclee:

CLEAR:TEMPORIZ=5:GOTO 7000[ENTER]

con lo que debería ver mostradas las horas de cinco temporizadores;

aunque, como aún no se ha introducido ninguno, la hora presentada en cada uno de ellos será «00-00-00» y no habrá ningún mensaje junto a ellas. La pulsación de cualquier tecla provocará la aparición del mensaje de error «Unexpected RETURN».

Módulo 1.3.10: Introducción de la hora en un temporizador de tiempo absoluto

Dentro de un momento introduciremos el módulo que nos permitirá ajustar los temporizadores, pero antes de que lo hagamos hay dos módulos más que son necesarios para que el usuario introduzca el tiempo de vencimiento de una de las dos formas diferentes. Cuando se vaya a ajustar un temporizador se le pedirá al usuario que especifique si la hora que va a introducir es un tiempo absoluto (es decir, la hora y minutos a los que debe sonar la alarma), o si es un tiempo de cuenta atrás (es decir, la alarma debe sonar *después* del número de horas y minutos especificado). Este módulo se ocupa de la introducción de tiempos absolutos y es similar al utilizado para ajustar la hora.

Módulo 1.3.10: Líneas 11000 - 11070

```
11000 REM*******************************
11010 REM Introduccion de datos en un
          temporizador absoluto
11020 REM*******************************
11030 INPUT #2, "Hora (0 a 23):",thora
11040 IF thora<0 OR thora>23 THEN PRINT
#2,"* * * * * LA HORA ESTA FUERA DEL INT
ERVALO: INTRODUZCA EL DATO OTRA VEZ* * *
 * *";:FOR i
=1 TO 2000:NEXT i:GOTO 11030
11050 INPUT #2,"Minutos (0 a 59):",tminu
to
11060 IF tminuto<0 OR tminuto>59 THEN PR
INT #2,"* * * *LOS MINUTOS ESTAN FUERA D
EL INTERVALO: INTRODUZCA EL DATO OTRA VE
Z* * * *";:F
OR i=1 TO 2000:NEXT i:GOTO 11050
11070 RETURN
```

Módulo 1.3.11: Introducción de la hora en un temporizador de cuenta atrás

Este segundo tipo de temporizador es inevitablemente más complicado ya que, en vez de tener que decir simplemente la hora a la que debe sonar la alarma, el programa tiene que calcular esa hora sumando la hora actual al período de tiempo que especifique el usuario.

Módulo 1.3.11: Líneas 10000 - 10090

```
10000 REM*******************************
10010 REM Introduccion de datos en un
          temporizador de cuenta atras
10020 REM*******************************
10030 INPUT #2, "Horas a transcurrir (0
a 23):",thora
10040 IF thora<0 OR thora>23 THEN PRINT
#2,"* * * * * LA HORA ESTA FUERA DEL INT
ERVALO: INTRODUZCA EL DATO OTRA VEZ* * *
 * *";:FOR i
=1 TO 2000:NEXT i:GOTO 10030
10050 INPUT #2,"Minutos a transcurrir (0
 a 59):",tminuto
10060 IF tminuto<0 OR tminuto>59 THEN PR
INT #2,"* * * *LOS MINUTOS ESTAN FUERA D
EL INTERVALO: INTRODUZCA EL DATO OTRA VE
Z* * * *";:F
OR i=1 TO 2000:NEXT i:GOTO 10050
10070 tminuto=tminuto+minuto:IF tminuto>
59 THEN tminuto=tminuto-60:thora=thora+1
10080 thora=thora+hora:IF thora>23 THEN
thora=thora-24
10090 RETURN
```

Comentarios

Líneas 10070-10080: El valor de la hora y los minutos actuales son calculados por el módulo de ajustes horarios que, cuando se haya introducido todo el programa, será llamado regularmente por una instrucción EVERY. Las horas y minutos introducidos por el usuario (THORA Y TMINUTO) pueden por tanto sumarse a las variables HORA y MINUTO.

Módulo 1.3.12: Ajuste de los temporizadores

Habiéndonos autocapacitado ya para introducir la hora de un temporizador, podemos pasar al módulo principal que permitirá al usuario la especificación de los ajustes de hasta 16 temporizadores. Observe que este módulo utiliza INPUT para obtener las tres unidades de información que necesita, por lo que mientras esté en funcionamiento los temporizadores no están siendo muestreados, y si algún temporizador llegara a activarse la alarma no sonará hasta que no se haya completado el módulo.

Observe también que no puede dejar a la máquina esperando indefinidamente la respuesta a la instrucción INPUT. Mientras está en este estado de espera, el sistema está registrando el número de veces que debería haberse efectuado la orden EVERY, número que en uno u otro momento se utilizará para ajustar la hora. Con el tiempo se agotará todo el espacio que se ha dejado para este propósito y, por cada EVERY que se pierda, se olvidará una llamada al módulo de sincronización, de forma que la hora presentada por el programa será incorrecta.

Módulo 1.3.12: Líneas 5000 - 5300

```
5000 REM*********************************
5010 REM Ajustar temporizador
5020 REM*********************************
5025 CLS #2:PRINT #2
5030 IF temporiz=16 THEN PRINT #2,"  **
 NO HAY TEMPORIZADORES LIBRES  **  ":FOR
 i=1 TO 2000:NEXT i:RETURN
5040 INPUT #2,"Cuenta atras (s/n)";atras
$
5050 IF LOWER$(atras$)="s" THEN GOSUB 10
000 ELSE GOSUB 11000
5060 ttemp=thora*60+tminuto
5070 rhor1=ttemp-hor:IF rhor1<=0 THEN rh
or1=rhor1+1440
5080 FOR i=0 TO temporiz-1
5090 rhor2=ttemp(i)-hor:IF rhor2<0 THEN
rhor2=rhor2+1440
5100 IF rhor1>rhor2 THEN NEXT i
5110 IF i<temporiz AND ttemp=ttemp(i) TH
EN PRINT #2,"****ESE TEMPORIZADOR YA EST
A AJUSTADO***":FOR j=1 TO 2000:NEXT:RETU
RN
5120 tnum=i
5130 mensaje$=""
```

```
5140 WHILE mensaje$=""
5150 INPUT #2,"Mensaje";mensaje$
5160 IF LEN(mensaje$)>25 THEN PRINT #2,"
***EL MENSAJE ES DEMASIADO LARGO. SU LON
GITUD MAXIMA DEBE SER DE 25 CARACTERES**
";:FOR i=1 T
O 2000:NEXT i:mensaje$=""
5170 WEND
5180 PRINT #2
5190 FOR i=temporiz-1 TO tnum STEP -1
5200 thora(i+1)=thora(i)
5210 tminuto(i+1)=tminuto(i)
5220 ttemp(i+1)=ttemp(i)
5230 mensaje$(i+1)=mensaje$(i)
5240 NEXT i
5250 temporiz=temporiz+1
5260 thora(tnum)=thora
5270 tminuto(tnum)=tminuto
5280 ttemp(tnum)=ttemp
5290 mensaje$(tnum)=mensaje$
5300 RETURN
```

Comentarios

Línea 5030: Si el número de temporizadores que ya se han ajustado es 16 (el máximo disponible) se genera un mensaje de error y el programa retorna al menú.

Líneas 5040-5050: Se le pide al usuario que especifique si desea un temporizador de cuenta atrás. La línea 5050 transforma la entrada, si fuera necesario, a letra minúscula con lo que puede entender tanto una «S» como una «s», después de lo cual llama a uno de los dos módulos anteriores.

Líneas 5060-5070: La hora proporcionada por el módulo de entrada horaria pertinente (el que se haya elegido de entre los dos anteriores) es transformada en una cifra que representa el número de minutos contados desde el comienzo del día. A esa cifra se le resta la hora actual expresada en minutos. Si el resultado es menor que cero, entonces se supone lógicamente que la hora de alarma especificada por el usuario es para el día siguiente, por lo que se le suman 1440 minutos (un día, expresado en minutos). Por ejemplo, si en el momento de ajustar el temporizador son las 18,30 (6,30 de la tarde) y el temporizador se ajusta a las 18.00 (6.00 de la tarde), entonces el programa supone que la alarma debe sonar a las 18.00 horas pero del día siguiente.

Líneas 5080-5120: La razón que justifica la presencia de este bucle es que el programa almacena los ajustes de todos los temporizadores

en el orden correcto, es decir, el primero en sonar es también el primero de la lista. Para conseguir este orden, la nueva información introducida debe ser comparada con las horas de los temporizadores que ya se han ajustado, y buscar consecuentemente el sitio de la lista en donde debe insertarse esta nueva información.

Si el primer temporizador/es de la lista tiene una hora inferior a la actual (es decir, se ha/n ajustado para que se active al día siguiente), entonces también hay que sumarle/s 1440 minutos a su hora para los fines de la comparación. Finalmente, puede encontrarse una hora posterior al ajuste del nuevo temporizador, en cuyo caso la sentencia IF de la línea 5100 hace que se termine el bucle FOR. Si no hay presente ningún ajuste posterior al nuevo, entonces el bucle continúa hasta que I es igual a TEMPORIZ, ya que cuando un bucle finaliza de forma natural, su variable es incrementada en uno en la última pasada (conforme la ejecución sale del bucle).

Por consiguiente, el valor de I apuntará ahora, o a un ajuste posterior al nuevo, o al primer espacio que haya tras el final de la lista actual de temporizadores. Se hace una última comprobación para ver si la posición indicada tiene un temporizador que ya esté ajustado a la hora que acaba de introducir el usuario. Si esto fuera así se generaría un mensaje de error y el programa retornaría al menú. Supuesto que no se encuentre tal error, la variable TNUM (el lugar en que es insertado el nuevo temporizador) se hace igual a I.

Líneas 5130-5170: Se solicita al usuario que introduzca el mensaje asociado al temporizador.

Líneas 5190-5240: Puesto que sabemos dónde se va a colocar el nuevo temporizador dentro de la lista, podemos mover una posición (desde esa posición en adelante) todos los temporizadores de dicha lista, dejando así un hueco. Cada temporizador necesita cuatro unidades de información para que sea registrado, y estas cuatro unidades son guardadas en las matrices THORA, TMINUTO, TTEMP y MENSAJE$.

Líneas 5250-5290: La variable TEMPORIZ es incrementada para indicar que se ha añadido otro temporizador, y los cuatro datos necesarios se insertan en las cuatro matrices en la posición indicada por TNUM.

Comprobación

En este estado podría comprobar el módulo ejecutando todo el programa. Cuando tenga el menú en pantalla especifique la opción 1 e introduzca:

S,0,1 y COMPROBACION

en respuesta a las cuatro peticiones, después de lo cual el programa debería retornar al menú. Ahora especifique la opción 3 para visualizar

los ajustes de los temporizadores, con lo que debería encontrar que al temporizador 0 se le ha dado un valor ligeramente inferior a un minuto por delante de la hora actual (la cantidad exacta depende de lo rápido que haya sido al introducir la información) y la etiqueta «COMPROBACION». No se moleste en esperar a que suene la alarma, puesto que aún no hemos introducido el módulo para lograr esto.

Módulo 1.3.13: Borrado de la pantalla

El único propósito de este programa es dejarlo correr en un segundo plano, de forma que pueda actuar como temporizador. Dejar conectado el CPC464 para conseguir esto es algo que no le causará ningún daño, pero dejar el monitor encendido y mostrando el mismo diseño durante largos períodos de tiempo puede dar lugar a que algunas de las partes más brillantes de lo que se está visualizando queden grabadas ligeramente en pantalla, y aparezcan permanentemente como imágenes fantasma sobre cualquier otra cosa que posteriormente se muestre en dicha pantalla (no se asuste, esto no es algo que vaya a ocurrir porque deje inadvertidamente su CPC464 y su monitor encendidos durante unas horas, estamos hablando de un uso continuado de un mismo diseño durante largos períodos).

Para solventar este posible problema, el programa tiene incorporado un módulo de protección de pantalla, módulo que la deja en blanco hasta que se pulse una tecla. Mientras la pantalla está en blanco, los temporizadores siguen siendo muestreados constantemente y cualquier alarma que se produzca sonará en la forma usual.

El módulo que nos ocupa se encarga de efectuar esta operación mediante el ajuste de todas las tintas al color de fondo. El módulo siguiente llama a éste, y crea un estado de espera hasta que la pantalla va a ser reactivada.

Módulo 1.3.13: Líneas 13000 - 13080

```
13000 REM******************************
13010 REM Apagado de la pantalla
13020 REM******************************
13030 INK 0,0
13040 INK 1,0
13050 INK 2,0
13060 INK 3,0
13070 BORDER 0
13080 RETURN
```

Módulo 1.3.14: Un estado de espera con una pantalla en blanco

Este módulo no es más que un bucle de espera del tipo de los que ya hemos visto en los módulos de menú y de visualización de los temporizadores, con el aditamento de que al ser llamado pone la pantalla en blanco y cuando se pulsa una tecla reinstaura los colores de comienzo.

Módulo 1.3.14: Líneas 8000 - 8130

```
8000 REM*********************************
8010 REM Borrar pantalla y esperar
8020 REM*********************************
8030 k$=""
8040 WHILE k$=""
8050 GOSUB 13000
8060 sonando=0
8070 WHILE k$="" AND sonando=0
8080 k$=INKEY$
8090 IF sonando=1 THEN k$=""
8100 WEND
8110 WEND
8120 GOSUB 12000
8130 RETURN
```

Comprobación

Ajuste uno o dos temporizadores para que suenen en breve, y llame desde el menú al estado de pantalla en blanco. Aunque no podrá ver el estado de los temporizadores, observará que las alarmas suenan en la forma normal.

Módulo 1.3.15: Anulación de un temporizador

A veces, puede darse el caso de que el usuario encuentre un temporizador ajustado a una hora incorrecta o que ya no se necesita. Este módulo brinda la oportunidad de poder suprimir tal temporizador. La técnica básica, aunque simple, es importante en programas de tratamiento de datos, y consiste en colapsar o hacer caer el fichero (la lista de temporizadores restantes) sobre el temporizador a suprimir.

Módulo 1.3.15: Líneas 6000 - 6140

```
6000 REM*********************************
6010 REM Anular temporizador
6020 REM*********************************
```

```
6025 CLS #2:PRINT #2
6030 IF temporiz=0 THEN PRINT #2,"***NO
HAY NINGUN TEMPORIZADOR AJUSTADO**":FOR
i=1 TO 2000:NEXT:RETURN
6040 INPUT #2,"Que temporizador desea an
ular";tnum
6050 IF tnum>temporiz OR tnum<=0 THEN PR
INT #2,"* * * * ESE TEMPORIZADOR NO EXIS
TE * * *":FOR i=1 TO 2000:NEXT:RETURN
6060 FOR i=tnum TO temporiz-1
6070 thora(i-1)=thora(i)
6080 tminuto(i-1)=tminuto(i)
6090 ttemp(i-1)=ttemp(i)
6100 mensaje$(i-1)=mensaje$(i)
6110 NEXT i
6120 temporiz=temporiz-1
6130 IF temporiz=0 THEN ttemp(0)=0
6140 RETURN
```

Comentarios
Líneas 6030-6050: Mensajes de error por si el usuario introduce un número de temporizador inexistente.
Líneas 6060-6110: El proceso de colapsamiento, que consiste en desplazar un espacio hacia abajo todo lo que está sobre el elemento a suprimir.

Comprobación
Ahora puede ajustar un temporizador, y llamar entonces a la opción 2 del menú, para borrarlo antes de que suene. Si la comprobación resulta satisfactoria el programa está listo para ser usado.

PROGRAMA 1.4: Cronómetro

Función del programa

El último programa de este capítulo de experimentos con horas es muy poco corriente, en el sentido de que trata de convertir a su 464 en un cronómetro. Obviamente, la precisión de un microordenador no es comparable a la de los relojes especializados; sin embargo, el ordenador tiene algunas ventajas como la de poder guardar un registro de los cronos que se han producido, realizar cálculos sobre ellos, etc. Este programa está diseñado para cronometrar uno o una

serie de sucesos, mostrar la serie de cronos correspondientes (con un mensaje o sin él), y para cada suceso calcular el período transcurrido desde el anterior. También puede, a petición, sacar por impresora la lista de cronos de forma que se pueda tener una copia permanente.

```
11:15:04 (00:00:04)
11:15:05 (00:00:01)
11:15:08 (00:00:03)
11:15:15 (00:00:07) AUTOBUS
11:15:24 (00:00:09)
11:15:25 (00:00:01)
11:15:27 (00:00:02)
11:15:27 (00:00:00)
11:15:35 (00:00:08) CAMION
11:15:37 (00:00:02)
11:15:38 (00:00:01)
11:15:40 (00:00:02)
11:15:48 (00:00:08) FURGONETA
11:15:50 (00:00:02)
11:15:51 (00:00:01)
11:15:54 (00:00:03)
11:15:57 (00:00:03)
11:16:02 (00:00:05) CARRO
```

Fig. 1.4 Vaciado del programa Cronómetro.

Los nuevos temas introducidos a lo largo de este programa son:

1) Cálculo de intervalos de tiempo.
2) Salida a una impresora.

Módulo 1.4.1: Inicialización

Un módulo estándar de inicialización que define una pequeña ventana implícita (la que se utilizará en caso de omisión) en la parte baja de la pantalla, una ventana de una única línea (#1) en la parte superior, y una segunda (#2) que abarca el área principal de la pantalla.

Módulo 1.4.1: Líneas 2000 - 2160

```
2000 REM********************************
2010 REM Inicializacion
2020 REM********************************
2030 MODE 1
```

```
2040 INK 0,1
2050 INK 1,24
2060 INK 2,26
2070 INK 3,2
2080 BORDER 1
2090 WINDOW 1,40,25,25
2100 PAPER 0:PEN 1:CLS
2110 WINDOW #1,1,40,1,1
2120 PAPER #1,0:PEN #1,2:CLS #1
2130 PRINT #1,"Ahora son las:"
2140 WINDOW #2,1,40,3,23
2150 PAPER #2,3:PEN #2,2:CLS #2:PRINT #2
2160 RETURN
```

Módulo 1.4.2: Ajuste inicial de la hora

El módulo estándar de ajuste inicial encontrado en los dos últimos programas.

Módulo 1.4.2: Líneas 3000 - 3090

```
3000 REM*******************************
3010 REM Ajuste de la hora
3020 REM*******************************
3030 CLS:PRINT "Ajuste la hora:":PRINT
3040 INPUT "Hora (1 a 12):";hora
3050 IF hora<1 OR hora>12 THEN PRINT "**
Fuera de intervalo : pruebe otra vez**":
GOTO 3040
3060 INPUT "Minutos (0 a 59):";minuto
3070 IF minuto<0 OR minuto>59 THEN PRINT
 "**Fuera de intervalo : pruebe otra vez
**":GOTO 3060
3080 segundo=0
3090 RETURN
```

Módulo 1.4.3: Esperando una entrada

Una vez más se utiliza INKEY$ para evitar los problemas asociados a EVERY cuando se utiliza INPUT. Además de crear el estado de espera, el módulo también es capaz de aceptar una instrucción para enviar salidas futuras a una impresora.

```
4000 REM*********************************
4010 REM Estado de espera
4020 REM*********************************
4030 t$="":mensaje$=""
4040 WHILE t$=""
4050 WHILE t$=""
4060 t$=INKEY$
4070 WEND
4080 WHILE LOWER$(t$)="i"
4090 impre=1-impre
4100 t$=""
4110 WEND
4120 WHILE t$=CHR$(13)
4130 INPUT "Mensaje (inferior a 18 carac
teres):",mensaje$
4140 IF LEN(mensaje$)<=18 THEN t$="*"
4150 WEND
4160 CLS
4170 WEND
4180 RETURN
```

Comentarios

Línea 4030: Esta cadena se utilizará para almacenar cualquier tecla pulsada por el usuario.

Líneas 4040-4170: El bucle principal, que estará iterando hasta que se le dé un valor a T$, ya sea por el programa, ya sea pulsando una tecla.

Líneas 4050-4070: El estado de espera en sí, que permanecerá hasta que se pulse una tecla.

Líneas 4080-4110: Si la tecla pulsada es la letra I, entonces el valor de la variable IMPRE bascula entre 0 y 1. Merece la pena tomar nota de esta simple línea, ya que representa la manera clásica de hacer que una variable permute entre dos valores. Si tiene una variable X y desea hacerla ir y venir entre los valores V1 y V2 (donde V1 es el más pequeño), lo puede conseguir de la manera siguiente:

X=V1+V2–X

En nuestro caso, dado que el valor inferior es cero, la expresión queda simplificada a:

X=V2–X

Líneas 4120-4150: Si se pulsa la tecla ENTER, entonces se pide al

usuario (en la línea de órdenes del fondo de la pantalla) que introduzca un breve mensaje para acompañar al crono correspondiente, después de lo cual la línea de órdenes es borrada de nuevo.

Comprobación

Asegúrese de que el programa está inicializado y entonces teclee:

GOTO 4000[ENTER]

con lo que el 464 entrará en el estado de espera. Si pulsa la tecla I no ocurrirá nada que sea visible. Si pulsa [ENTER] se le pedirá que introduzca un mensaje y terminará el estado de espera. La pulsación de cualquier otra tecla hace que el módulo finalice.

Módulo 1.4.4: Ajustes horarios

Este módulo es similar al de los dos programas anteriores con la excepción de que, aparte de actualizar constantemente el tiempo, también actualiza una segunda forma de tiempo que representa el intervalo transcurrido desde la última vez que se pulsó una tecla. Esta segunda forma se almacena en variables que comienzan siempre por la letra P.

Módulo 1.4.4: Líneas 6000 - 6280

```
6000 REM*******************************
6010 REM Ajustes horarios
6020 REM*******************************
6030 segundo=segundo+1
6040 WHILE segundo>59
6050 segundo=0:minuto=minuto+1
6060 WHILE minuto>59
6070 minuto=0:hora=hora+1
6080 WHILE hora>12
6090 hora=1
6100 WEND
6110 WEND
6120 WEND
6130 psegundo=psegundo+1
6140 WHILE psegundo>59
6150 psegundo=0:pminuto=pminuto+1
6160 WHILE pminuto>59
6170 pminuto=0:phora=phora+1
6180 WHILE phora>12
```

```
6190 phora=1
6200 WEND
6210 WEND
6220 WEND
6230 LOCATE#1,16,1
6240 h=hora:m=minuto:s=segundo:GOSUB 700
0
6250 PRINT#1,hor$;
6260 h=phora:m=pminuto:s=psegundo:GOSUB
7000
6270 PRINT#1," (";hor$;")"
6280 RETURN
```

Comprobación

El módulo no puede comprobarse de forma eficaz hasta que no se introduzca el siguiente, que permitirá la presentación de la hora.

Módulo 1.4.5: Presentación de los resultados

Este módulo, como en los dos programas anteriores, genera una cadena a partir de la hora actual.

Módulo 1.4.5: Líneas 7000 - 7050

```
7000 REM*********************************
7010 REM Creacion de una cadena horaria
7020 REM*********************************
7030 hor$=STR$(1000000+h*10000+m*100+s)
7040 hor$=MID$(hor$,3,2)+":"+MID$(hor$,5
,2)+":"+MID$(hor$,7,2)
7050 RETURN
```

Comprobación

Convierta temporalmente la línea 6280 en:

6280 END

y teclee:

CLEAR:CLS:GOTO 6000[ENTER]

con lo que debería ver:

00:00:01 (00:00:01)

en la parte superior de la pantalla. Si teclea GOTO 6000 repetidas veces, verá cómo la hora se incrementa en 1 segundo cada vez.

No se olvide de poner la línea 6280 en su estado original.

Módulo 1.4.6: Sacando los resultados por la impresora

Habiendo introducido los módulos que ponen el programa en estado de espera y que le permiten actualizar el tiempo, necesitamos ahora poder imprimir el tiempo calculado cuando se pulsa una tecla.

Módulo 1.4.6: Líneas 5000 - 5100

```
5000 REM*******************************
5010 REM Mostrar o imprimir valores
5020 REM*******************************
5030 h=hora:m=minuto:s=segundo:GOSUB 700
0
5040 p$=" "+hor$
5050 h=phora:m=pminuto:s=psegundo:GOSUB
7000
5060 p$=p$+" ("+hor$+") "+mensaje$
5070 PRINT#2,p$
5080 IF impre=1 THEN PRINT#8,p$
5090 phora=0:pminuto=0:psegundo=0
5100 RETURN
```

Comentarios

Líneas 5030-5070: Se construye una cadena que consta de la hora actual (HORA, MINUTO, SEGUNDO), más el intervalo de tiempo (PHORA, PMINUTO, PSEGUNDO), más cualquier mensaje que haya introducido el usuario. La presentación se hace por la ventana principal de la pantalla, la (#2).

Línea 5080: Al contrario que muchos otros micros, el 464 mantiene permanentemente abierto un canal de comunicación con cualquier impresora conectada. En la mayoría de los otros micros sería necesario «abrir un fichero para la impresora», cosa que requiere varias líneas de BASIC. En el 464, todo lo que se necesita es enviar lo que queramos imprimir por el número de canal 8, que siempre está conectado directamente al port de la impresora.

Línea 5090: Los valores del intervalo se ponen a 0, ya que sólo tienen el propósito de representar el tiempo transcurrido desde la última tecla pulsada.

Módulo 1.4.7: El módulo de control

El toque final es el módulo de control, que convierte al programa en una unidad de trabajo.

Módulo 1.4.7: Líneas 1000 - 1110

```
1000 REM*********************************
1010 REM Control
1020 REM*********************************
1030 ON BREAK GOSUB 1110
1040 GOSUB 3000
1050 EVERY 50 GOSUB 6000
1060 GOSUB 2000
1070 WHILE 1
1080 GOSUB 4000
1090 GOSUB 5000
1100 WEND
1110 CLEAR:MODE 1:END
```

Comentarios
Línea 1070: Un pequeño toque que puede encontrar útil: el «1» que hay tras WHILE permite que el bucle continúe indefinidamente. Cualquier valor positivo cumpliría la misma función.

Conclusión

Hay un cierto número de enseñanzas a sacar de los programas de este capítulo, no siendo la menor de ellas el que los usos de su 464 solamente están limitados por su imaginación. Pero, quizás, la lección más importante que puede aprender, si vuelve a remirar los programas, es lo fácil que resulta el diseño de programas cuando todo está contenido en módulos ordenados que se pueden transferir de un programa al siguiente. Con su orden MERGE, cuyo uso no puede ser más fácil, el 464 pide a gritos ser utilizado de esta forma. Una vez que se haya construido una biblioteca suficiente de rutinas útiles, encontrará que la mayor parte de su programación es como montar un rompecabezas, excepto que el programa terminado será, al menos a veces, más útil.

2. Pintando con números

En este capítulo dejaremos de jugar con las horas para pasar a otra cosa que el 464 hace particularmente bien: presentar la información en forma más rápidamente entendible que las listas de datos y de cifras. El capítulo consta de tres programas que crearán gráficos de una u otra clase, tanto en alta como en baja resolución.

Conforme avancemos en el capítulo, y en los capítulos que siguen, encontrará que las explicaciones, excepto cuando se presentan nuevas ideas o módulos complejos, se van haciendo más breves. Se trata de algo hecho deliberadamente, para intentar conseguir un equilibrio entre la información que necesita para entender los programas que está introduciendo, y los programas en sí, que al fin y al cabo son la razón por la que compró el libro. *Si* en algún momento se *encontrara* desconcertado, o perdido, verá que por lo general siempre puede retroceder a un programa anterior en el que se ha utilizado una técnica similar, y releer los comentarios antes de continuar (una vez aclarada la cuestión). Esta es la razón por la que se le recomendó en la Introducción que fuera avanzando poco a poco por el libro en vez de saltarse a alguno de los programas más ambiciosos del final.

Los programas incluidos en este capítulo son:

GRAFICOS: Que crea gráficos lineales muy eficaces y potentes en alta resolución.

GRAFICO-SECTORES: Que tomará una cantidad finita de datos y los mostrará bajo la forma de un círculo multicolor dividido en sectores circulares.

GRAFICOS-3D: Que le permitirá crear un sorprendente gráfico de barras en tres dimensiones (3D).

PROGRAMA 2.1: Gráficos

Función del programa

Uno de los problemas que consumen más tiempo y memoria, en los programas que trabajan con datos, es la forma en que los datos deben ser introducidos. A menudo se necesitan secciones de programa bastante largas y complejas dedicadas a la entrada, alteración y supresión de datos.

En los dos primeros programas de este capítulo examinaremos la forma de rodear esta limitación, creando un programa que es fácil de usar, pero que no tiene que dedicar memoria para presentar peticiones en pantalla, almacenar datos en matrices o arreglar los datos para grabarlos en cinta. Los programas interactivos, es decir, los programas que piden información al usuario conforme se van ejecutando, son ciertamente agradables de usar, pero pueden ser un lujo cuando la memoria es escasa. En este programa, que traza un gráfico en alta resolución, hacemos uso de sentencias DATA para crear un programa que es sencillo de utilizar, y aún más sencillo de escribir que un programa interactivo que consiguiera el mismo objetivo.

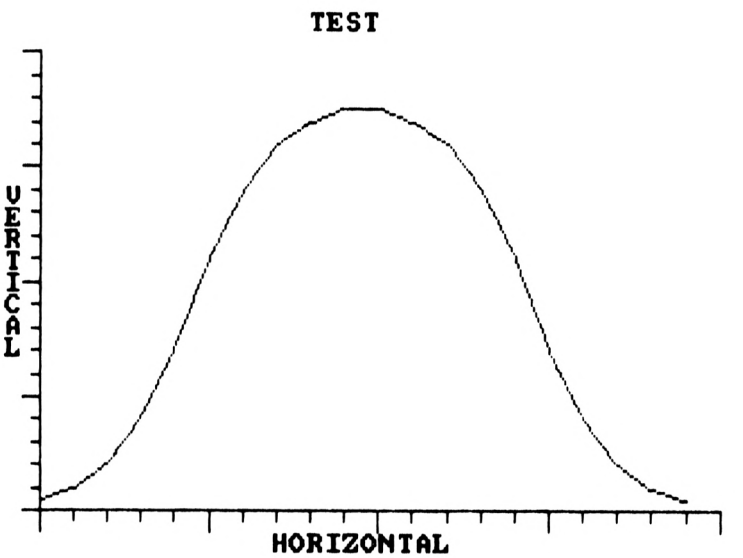


Fig. 2.1 Visualización en pantalla del programa Gráficos.

Las técnicas introducidas a lo largo del programa incluyen:

1) El uso flexible de las sentencias DATA.

Módulos 2.1.1 y 2.1.2: Los datos para el gráfico

Estos dos módulos son la clave de la simplicidad del programa. En estas líneas está contenida toda la información que será necesaria para trazar un gráfico satisfactorio, toda ella claramente clasificada y rotulada para que el usuario no tenga ningún tipo de dificultad en la preparación de la presentación en pantalla. No se preocupe si en este momento encuentra dificultad en ver el sentido o la pertinencia de todos los números; su uso se hará evidente una vez haya introducido y ejecutado el programa una o dos veces, y haya experimentado haciendo algunos cambios.

Módulos 2.1.1 y 2.1.2: Líneas 3000 - 4050

```
3000 REM*******************************
3010 REM Datos 1
3020 REM*******************************
3030 DATA TITULO DEL GRAFICO,TEST
3040 DATA NOMBRE DEL EJE H,HORIZONTAL
3050 DATA NOMBRE DEL EJE V,VERTICAL
3060 DATA DIVISIONES DEL EJE H,20
3070 DATA DIVISIONES DEL EJE V,20
3080 DATA CANTIDAD POR DIVISION DE EJE V
,10
4000 REM*******************************
4010 REM Datos 2
4020 REM*******************************
4030 DATA 5,10,20,40,70,110,140,160,170,
175
4040 DATA 175,170,160,140,110,70,40,20,1
0,5
4050 DATA FINAL
```

Comentarios

Líneas 3030-3050: El nombre que el usuario desea dar a la totalidad del gráfico, y los rótulos a poner junto a los ejes vertical y horizontal. Observe que en cada caso la frase que hay antes de la coma de la sentencia data, está allí únicamente para conveniencia del usuario, ya que será ignorada por el programa; la coma es esencial para separar la primera frase de la información importante que le sigue.

Líneas 3060-3070: Los dos ejes del gráfico serán fraccionados en divisiones, para facilitar la lectura. Los ejes tendrán siempre la misma longitud, pero el usuario puede especificar cuántas divisiones hay que poner en cada uno de ellos.

Línea 3080: Si, por ejemplo, el gráfico registrara las toneladas de trigo

producidas por un país a lo largo de un cierto número de años, es posible que el usuario pueda desear que cada división del eje vertical represente 1000 toneladas; en vez de obligar al usuario a que divida la cifra real por 1000 antes de introducirla, la cifra de esta línea permite que se especifique la cantidad por división, de forma que cada cifra se puede introducir en su valor real.
Líneas 4030-4040: Los datos sobre los que estará basado el gráfico. En el caso concreto de este programa, estos números generarán una curva uniforme en forma de campana. Observe que solamente se han tomado medidas para una única cifra por cada división del eje horizontal, comenzando por la posición uno, aunque no hay porqué utilizar la totalidad de dicho eje. Se pueden incluir tantos datos individuales como se desee en cada línea particular.
Línea 4050: Puede utilizar tantas sentencias data como le permita la memoria, pero la información debe terminar con una sentencia data que contenga la palabra FINAL (o algo similar). Esta es la señal utilizada para dar a entender al programa que ha llegado al final de los datos a utilizar para el gráfico (aún cuando detrás sigan más sentencias data).

Módulo 2.1.3: Trazado de los ejes y divisiones del gráfico

Este módulo dibuja la plantilla bajo la que se colocará el gráfico final, aderezada con las divisiones de los ejes y los diversos rótulos especificados.

Módulo 2.1.3: Líneas 1000 - 1190

```
1000 REM*******************************
1010 REM Trazar ejes,divisiones...
1020 REM*******************************
1030 MODE 1
1040 MOVE 32,368
1050 DRAW 32,32,2
1060 DRAW 639,32
1070 RESTORE 3000
1080 READ t$,t$:LOCATE 20-LEN(t$)/2,1:PR
INT t$
1090 READ t$,t$:LOCATE 21-LEN(t$)/2,25:P
RINT t$
1100 READ t$,t$:FOR i=1 TO LEN(t$):LOCAT
E 1,12-LEN(t$)/2+i:PRINT MID$(t$,i,1):NE
XT i
1110 READ t$,nv:lv=338/nv
1120 FOR i=0 TO nv
```

```
1130 MOVE 24+8*(i/5=INT(i/5)),32+i*lv:DR
AW 32,32+i*lv
1140 NEXT i
1150 READ t$,nh:lh=606/nh
1160 FOR i=0 TO nh
1170 MOVE 32+i*lh,24+8*(i/5=INT(i/5)):DR
AW 32+i*lh,32
1180 NEXT i
1190 READ t$,unidad
```

Comentarios

Líneas 1040-1060: Se trazan los dos ejes: una línea desde cerca de la esquina superior izquierda de la pantalla hasta cerca de la esquina inferior izquierda, continuando a 90 grados con otra línea que sigue la parte inferior de la pantalla, hasta un lugar próximo a la esquina inferior derecha.

Línea 1070: El «puntero de datos» del 464 se pone apuntando hacia el primer dato que hay tras el comienzo del módulo de la línea 3000. La instrucción RESTORE impedirá que se genere un error «DATA exhausted» (no hay más DATOS) en caso de que el programa se arranque con un GOTO. De cualquier modo, la utilización de un RUN colocará el puntero automáticamente apuntando hacia el primer dato.

Líneas 1080-1100: Los rótulos del gráfico como un todo. Los correspondientes a los ejes horizontal y vertical se leen de las sentencias data y se presentan en pantalla. En el caso del rótulo del eje vertical se utiliza un bucle para presentarlo con cada carácter debajo del anterior, y en el lado izquierdo de la pantalla. Observe que en cada caso hay dos sentencias READ; la primera extrae la frase que hay *antes* de la coma de la sentencia data, frase que es descartada inmediatamente mediante la lectura e introducción de otra cadena en la misma variable, la T$.

Línea 1110: El número de divisiones a poner en el eje vertical (NV) se lee en la sentencia data siguiente. La longitud del eje vertical (338 pixels) se divide entonces por este número, llegándose así a la longitud de cada división en pixels (LV).

Líneas 1120-1140: Este bucle traza pequeñas marcas en el eje vertical, para registrar las divisiones especificadas por el usuario. Algo a tener especialmente en cuenta en estas líneas es la expresión 8*(I/5=INT(I/5)), que tiene el efecto de hacer que las marcas de las divisiones múltiplos de 5 sean mayores que el resto. Para entender la expresión, necesita saber algunas cosas sobre la forma en que son tratadas las «condiciones lógicas» por el 464.

Cuando el intérprete BASIC del 464, el imponente programa en código máquina que ejecuta el BASIC por Vd, se tropieza con una

condición que va detrás de un IF (algo como «A>B», «A=B» o «A<B»), necesita determinar si la condición es verdadera o falsa antes de decidir si llevar a cabo la acción especificada por la sentencia IF. Así:

```
IF A>B THEN GOSUB 1000
```

se actuará si «A>B» es *verdadero* (por ejemplo: A=10 y B=9), o se ignorará si es falso (por ejemplo: A=9 y B=10). Para tomar la decisión, la condición es evaluada de acuerdo con los valores actuales de A y B (o cualesquiera otras variables se hayan especificado), y se le asigna un valor que es –1 en caso de que sea verdadera, o 0 si la condición es falsa. Más que la condición en sí, lo que importa es el *valor*, un hecho que puede demostrarse a sí mismo si pone en práctica la prueba siguiente. Introduzca en modo directo:

```
A=5[ENTER]
IF A THEN PRINT «VERDADERO»[ENTER]
```

el resultado será que se presenta «VERDADERO» en pantalla, ya que el valor que sigue a la sentencia IF no es 0 (cualquier valor distinto de cero, sea positivo o negativo, producirá el mismo resultado). Ahora pruebe con:

```
A=0[ENTER]
IF A THEN PRINT «VERDADERO»[ENTER]
```

y verá que esta vez no se presenta nada en pantalla. Por el momento, sin embargo, no estamos demasiado interesados en la forma en que IF trabaja, sino en la forma en que son evaluadas las condiciones, así que pruebe con lo siguiente:

```
A=1[ENTER]
B=1[ENTER]
PRINT A=B[ENTER]
```

lo que debería ver es «–1», el valor de una condición verdadera. Ahora pruebe con:

```
A=1[ENTER]
B=2[ENTER]
PRINT A=B[ENTER]
```

El resultado será 0 ya que la condición no es verdadera. En principio, esto puede parecer interesante pero un poco fuera de lugar. En

realidad, esta capacidad de extraer un valor a partir de una condición lógica es de gran valor en programación, según queda ilustrado en la línea 1130.

Lo que hace la línea 1130 es trazar una serie de líneas perpendiculares al eje vertical, para marcar las divisiones especificadas por el usuario (la longitud de estas líneas es normalmente de ocho pixels). Sin embargo, siempre que la variable de bucle I es divisible por 5 (es decir, cada cinco marcas), la condición (I/5=INT(I/5)) será verdadera y tomará el valor –1 en vez de 0. En otras palabras, la inclusión de «–8*(I/5=INT(I/5))» en la línea que especifica la longitud de la marca a trazar (mediante DRAW), permite que cada cinco marcas se doble la longitud sin tener que usar la compleja sentencia IF...THEN...ELSE.

Observe que para *sumar* ocho a la longitud de la marca, tenemos que *restar* ocho veces el valor de la condición, ya que cuando es verdadera su valor es *menos* uno (restar un número negativo es equivalente a sumar uno positivo).

Líneas 1150-1180: Se lleva a cabo exactamente el mismo proceso, pero para el eje horizontal.

Líneas 1190: Se lee, en la sentencia data, el número de unidades representadas por cada división del eje vertical.

Comprobación

Para comprobar esta parte del programa, todo lo que se necesita es correrlo hasta el punto en que nos encontramos. Debería ver trazados los ejes con la palabra «TEST» en la parte superior de la pantalla, «VERTICAL» en la parte izquierda y «HORIZONTAL» a lo largo de la parte inferior. Los dos ejes deberían estar claramente fraccionados en 20 divisiones.

Módulo 2.1.4: Trazado de la curva

Habiendo preparado la plantilla, todo lo que queda es dibujar la curva en sí, utilizando para ello la información especificada en el módulo DATOS 2.

Módulo 2.1.4: Líneas 2000 - 2160

```
2000 REM*******************************
2010 REM Trazar curva
2020 REM*******************************
2030 ON ERROR GOTO 2140
2040 RESTORE 4000
2050 READ t
2060 PLOT 32,32+t/unidad*lv,1
2070 columna=1
```

```
2080 READ t$
2090 IF t$="FINAL" THEN GOTO 2130
2100 DRAW 32+lh*columna,32+VAL(t$)/unida
d*lv
2110 columna=columna+1
2120 GOTO 2080
2130 IF INKEY$="" THEN GOTO 2130
2140 CLS
2150 LIST 3000-
2160 END
```

Comentarios

Líneas 2030 y 2130-2160: Si se genera un error durante la secuencia principal de trazado, o se pulsa una tecla una vez que se ha trazado la curva, la pantalla se borrará y se presentarán los datos en los que estaba basado el gráfico. Esto hace sumamente fácil el examen del gráfico y el cambio de los datos.

Línea 2050: Se extrae el primer dato.

Línea 2060: Se comienza el gráfico en el eje vertical, en un punto que representa la altura del primer dato (expresada en las unidades especificadas por el usuario).

Línea 2070: La variable COLUMNA se utilizará para registrar el número de datos que se han leído y, por tanto, cuánto se ha movido el gráfico en el sentido del eje horizontal.

Líneas 2080-2090: Los datos posteriores son leídos como cadenas. Esto nos permite comprobar si el dato leído es la palabra «FINAL», para terminar el programa en caso afirmativo.

Líneas 2100-2120: Puesto que el cursor de gráficos está ya situado al final de la parte del gráfico trazado, todo lo que se necesita hacer es trazar (DRAW) una línea hasta el siguiente punto, incrementando el valor de COLUMNA por cada dato que se vaya leyendo. La coordenada «X», u horizontal, se calcula multiplicando el valor de COLUMNA (el número de divisiones que ha avanzado el gráfico a lo largo del eje horizontal) por la longitud en pixels de las divisiones horizontales (LH); la constante 32 es la distancia que hay entre el comienzo del eje X y el borde izquierdo de la pantalla. Las coordenadas «Y», o verticales, son en primer lugar divididas por UNIDAD.

Así, si el dato fuera 1.000.000 y el usuario hubiera especificado que el eje vertical tenía que ser dividido en unidades de 100.000 (UNIDAD=100.000), entonces el resultado sería 1.000.000/100.000 o 10 divisiones. Habiendo llegado al número de divisiones, este número es multiplicado por la longitud en pixels de cada división vertical (LV).

Comprobación

Lance el programa y debería ver trazada una curva uniforme en

forma de campana. Cuando se haya completado el dibujo pulse cualquier tecla y debería ver listados en pantalla los módulos DATA, permitiéndole alterarlos a voluntad. Si esta prueba es satisfactoria, el programa está listo para ser utilizado.

PROGRAMA 2.2: Gráfico-sectores

Función del programa

Una forma muy útil de presentar pequeñas cantidades de datos es la técnica de los gráficos de sectores en la que un círculo, que representa la suma total, es dividido en sectores que representan las diferentes partes que componen el total. En el programa que viene a continuación, recurriremos a lo que ya hemos aprendido acerca de las matemáticas de una circunferencia y del uso flexible de la sentencia data en el último capítulo. Si alguno de esos temas no lo tiene muy claro retroceda y échele otro vistazo, ya que no tendremos tiempo de explicar otra vez los detalles de ambas técnicas.

En este caso, no hay necesidad de suprimir las líneas que se introducen durante las comprobaciones ya que estas líneas formarán parte del último módulo, el módulo de control.

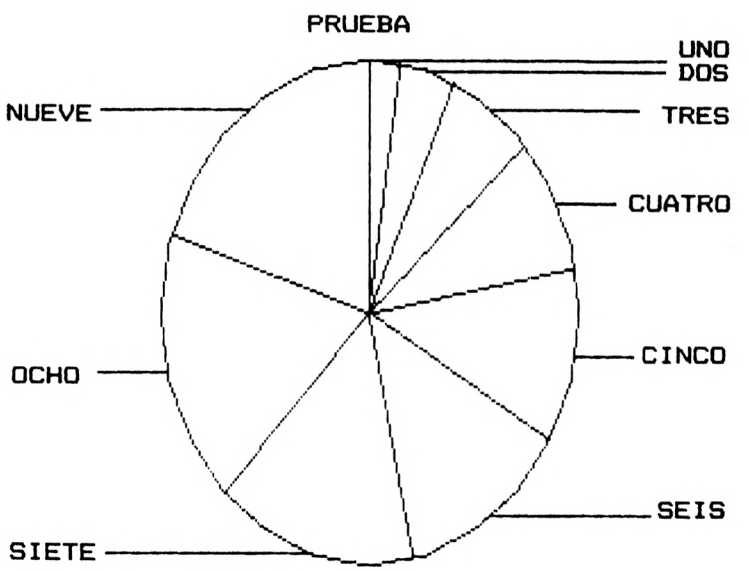


Fig. 2.2 Visualización del programa Gráfico-sectores.

Módulo 2.2.1: Los datos del gráfico

Como en el anterior gráfico de alta resolución, los números en que se basará el gráfico que nos ocupa están contenidos en sentencias data que se explican por sí mismas. Observe sin embargo que, tal como está listado el programa, las dos matrices que se utilizarán para guardar el nombre de cada concepto y su valor no están dimensionadas, así que está limitado a un máximo de 10 conceptos diferentes. Sinceramente, un gráfico de sectores con más de diez conceptos tiene poco valor, ya que estará demasiado apiñado como para dar cabida claramente a toda la información. Aún así, puede incluir, si lo desea, una sentencia de dimensionado al comienzo del programa para incrementar el número de conceptos que puedan manejarse.

Módulo 2.2.1: Líneas 4000 - 4080

```
4000 REM*********************************
4010 REM Datos para la grafica
4020 REM*********************************
4030 DATA TITULO,PRUEBA
4040 DATA NUMERO DE ELEMENTOS,9
4050 DATA NOMBRES DE LOS ELEMENTOS,UNO,D
OS,TRES,CUATRO,CINCO,SEIS,SIETE,OCHO,NUE
VE,DIEZ
4060 DATA
4070 DATA CANTIDAD POR ELEMENTO,1,2,3,4,
5,6,7,8,9,10
4080 DATA
```

Módulo 2.2.2: Proceso de los datos del gráfico

La información contenida en el módulo DATA es leída y transferida a las variables NOMBRE$ y ARTI, y a las matrices NOMBRE$ y A.

Módulo 2.2.2: Líneas 5000 - 5110

```
5000 REM*********************************
5010 REM Tratamiento de los datos
5020 REM*********************************
5030 RESTORE 4000
5040 READ t$,nombre$
5050 READ t$,arti
5060 READ t$:FOR i=0 TO arti-1:READ nomb
re$(i):NEXT i
```

```
5070 RESTORE 4070
5080 suma=0:READ t$:FOR i=0 TO arti-1:RE
AD t:suma=suma+t:NEXT i
5090 RESTORE 4070
5100 READ t$:FOR i=0 TO arti-2:READ t:a(
i+1)=(t/suma)*360+a(i):NEXT i
5110 RETURN
```

Comentarios

Líneas 5080-5100: En primer lugar se suma el valor de todos los artículos o conceptos para calcular el total que debe representar el círculo. El puntero de las sentencias DATA es colocado entonces (mediante RESTORE) al comienzo de los números que representan la cantidad de cada artículo, y cada una de esas cantidades es transformada en una segunda, que al ser dividida por 360 daría el mismo resultado que el cociente entre la cantidad original y el total. Por ejemplo, si el total fuera 100 y la cantidad de uno de los artículos fuera 25, este segundo número sería transformado en 90 (el 25 % de 360). Estos nuevos números se utilizarán posteriormente para determinar el tamaño del sector circular que se debe asignar a cada artículo.

Comprobación

Introduzca las líneas que se presentan a continuación (que son parte de lo que finalmente será el módulo de control) y ejecute, entonces, lo que va de programa:

```
1040 GOSUB 5000
1100 END
```

Si todo va bien, no debería ocurrir nada (nada visible), solamente si hay un error de algún tipo podrá ver algo. No obstante, si así lo desea, puede presentar los valores de las variables y de las matrices que aparecen en el módulo para tener una mayor seguridad.

Módulo 2.2.3: Preparación de la pantalla

Este módulo prepara el modo de gráficos y los colores asociados, y traza una circunferencia de radio 184 localizada en el centro de la pantalla, junto con el nombre dado a la gráfica.

Módulo 2.2.3: Líneas 2000 - 2090

```
2000 REM*******************************
2010 REM Trazado de la estructura
2020 REM*******************************
2030 MODE 1:ORIGIN 320,184:BORDER 0
2040 INK 0,0:INK 1,2:INK 3,18
2050 LOCATE 20-LEN(nombre$)/2,1:PEN 2:PR
INT nombre$
2060 DEG
2070 PLOT 0,184,3
2080 FOR a=0 TO 360 STEP 15:DRAW 184*SIN
(a),184*COS(a):NEXT a
2090 RETURN
```

Comentarios
Línea 2080: Si lee las explicaciones sobre las matemáticas de una circunferencia, dadas en los comentarios del primer programa del libro (cap. 1), se dará cuenta de que lo que hace esta línea es trazar simplemente una circunferencia. En vez de pintar cada punto individual de la circunferencia, lo que hacemos aquí es calcular una serie de puntos que están separados 15 grados entre sí y trazar posteriormente líneas que los unen.

Comprobación
Añada las líneas siguientes y después lance el programa:

```
1060 GOSUB 2000
1080 IF INKEY$=«»THEN GOTO 1080
```

el resultado no debería ser nada más emocionante que el título dado a la gráfica y una circunferencia amarilla. Pulse cualquier tecla que no sea ESC para volver a la pantalla normal.

Módulo 2.2.4: Introducción de los detalles

Este módulo traza los sectores en que será dividido el círculo, los colorea y les asigna los rótulos especificados en el módulo DATA. Para comprender lo que está ocurriendo necesitará recordar las matemáticas de un círculo explicadas en el programa Analoj. Si las ha olvidado, debería retroceder y volver a echar un vistazo a los comentarios de ese programa.

```
3000 REM*******************************
3010 REM Trazado de los sectores
3020 REM*******************************
3030 FOR i=0 TO arti-1
3040 MOVE 0,0:DRAW 184*SIN(a(i)),184*COS
(a(i))
3050 NEXT i
3055 PEN 1
3060 FOR i=0 TO arti-1
3070 ta=((a(i)+a(i+1+arti*(i+1=arti))))/
2
3080 IF a(i+1+arti*(i+1=arti))<a(i) THEN
 ta=ta+180
3090 MOVE 180*SIN(ta),180*COS(ta):c=i MO
D 3+1:IF i=arti-1 AND c=1 THEN c=2
3100 GOSUB 6000
3110 tx=184*SIN(ta)
3120 ty=184*COS(ta)
3130 dx=320*SGN(tx)
3140 MOVE tx,ty:DRAW dx,ty,3
3150 LOCATE 1+(LEN(nombre$(i))-40)*(dx=3
20),14-INT((ty+9)/16):PRINT nombre$(i);
3160 NEXT i
3170 RETURN
```

Comentarios

Líneas 3030-3050: Se trazan una serie de líneas desde el centro del círculo hasta su perímetro, líneas que dividen el círculo en los sectores de la gráfica. Los números utilizados son los calculados en el módulo 2.2.2.

Líneas 3070-3100: Se calculan más ángulos, pero esta vez sus posiciones están en el medio de los sectores que se acaban de trazar. La expresión lógica de las líneas 3070 y 3080 (ARTI*(I+1=ARTI)), asegura que, cuando se ha alcanzado el último sector, el comienzo del primer sector se utiliza como el segundo ángulo para calcular la posición media. Al hacer este último cálculo el resultado será erróneo: en vez de sumar un ángulo con un ángulo mayor y dividir por 2 para encontrar el punto medio, el ángulo del último sector será sumado a 0 (el comienzo del primer sector), produciendo así un resultado de un ángulo que está a medio camino entre el comienzo del primero y el último sector. Esto se corrige añadiendo 180 grados al último resultado.

La línea 3090 calcula entonces, en base a este ángulo, una po-

sición que está dentro del perímetro del círculo. Una llamada al módulo siguiente, que es muy similar al módulo de «coloreado» utilizado en Analoj, se utiliza entonces para dar color al sector (de forma de cuña) en donde está situado el punto. La línea 3090 genera también una sucesión cíclica de los tres colores de primer plano especificados en el módulo de inicialización, exceptuando al último sector que no puede tener el mismo color que el primero. La razón de ello es que, puesto que ambos sectores están uno junto al otro, la comprensión del gráfico sería difícil si ambos tuvieran el mismo color.
Líneas 3110-3120: Estas dos líneas calculan la posición de un punto situado sobre la periferia del círculo y a mitad de camino entre los puntos primero y último del sector en cuestión.
Línea 3130: Al preparar la plantilla del gráfico, hemos movido el origen de las órdenes de gráficos al centro de la pantalla. Esta línea calcula una posición de la pantalla situada a 320 pixels del centro. La variable TX ya guarda la coordenada X de un punto de la circunferencia que puede ser negativa (a la izquierda del centro) o bien positiva (a la derecha del centro). Al multiplicar 320 por SGN(TX) nos aseguramos de que si el centro del sector está a la izquierda del centro de la pantalla, también lo estará DX (y viceversa).
Línea 3140-3150: Se traza una línea desde la periferia del círculo hasta el borde izquierdo o derecho (según lo indique DX) de la pantalla. Al final de la línea, o más bien sobre ella, se imprime el rótulo del sector al que apunta la línea. La posición de presentación de los rótulos del lado derecho se mueve hacia la izquierda (de forma que no se salgan fuera de pantalla) utilizando una condición lógica para determinar la coordenada X. La línea parece más compleja de lo que realmente es, todo por la necesidad de transformar la posición, previamente calculada en pixels, a una posición para caracteres.

Comprobación

Añada las líneas siguientes y lance el programa:

```
1070 GOSUB 3000
6000 RETURN
```

Debería ver una figura como la que está al comienzo de este programa, aunque los sectores aún no estarán coloreados.

Módulo 2.2.5: Coloreado de los sectores

Este módulo es casi idéntico al contenido en el programa Analoj. La única diferencia es que el proceso de coloreado continúa hasta que encuentra cualquier color distinto del color de fondo.

Módulo 2.2.5: Líneas 6000 - 6300

```
6000 REM*******************************
6010 REM Coloreado de los sectores
6020 REM*******************************
6030 IF TESTR(0,0)<>0 THEN RETURN
6040 s=2
6050 MOVE s*INT(XPOS/s),2*INT(YPOS/2)
6060 :
6070 :
6080 REM buscar hacia arriba/derecha***
6090 IF TESTR(0,s)=0 THEN GOTO 6090
6100 IF TESTR(s,-s)=0 THEN GOTO 6090
6110 :
6120 :
6130 REM buscar hacia arriba/izquierda*
6140 MOVER -s,0
6150 IF TESTR(0,s)=0 THEN GOTO 6090
6160 IF TESTR(-s,-s)=0 THEN GOTO 6150
6170 :
6180 :
6190 REM colorear segun lineas horizon-
       tales************************
6200 x1=XPOS
6210 MOVER s,0
6220 PLOTR 0,0,c
6230 IF TESTR(s,0)=0 THEN GOTO 6220
6240 x2=XPOS-s
6250 MOVE x1,YPOS-s
6260 IF TESTR(s,0)=0 THEN GOTO 6290
6270 IF XPOS=x2 THEN RETURN
6280 GOTO 6260
6290 IF TESTR(-s,0)<>0 THEN GOTO 6200
6300 GOTO 6290
```

Comprobación

Igual procedimiento que con el módulo anterior. En este caso, los sectores estarán coloreados.

Módulo 2.2.6: El módulo de control

La mayoría de las líneas de este módulo ya se han introducido, pero asegúrese de que tiene todas las líneas que se listan más abajo o le faltarán uno o dos refinamientos.

Módulo 2.2.6: Líneas 1000 - 1120

```
1000 REM********************************
1010 REM Control
1020 REM********************************
1030 ON ERROR GOTO 1110
1040 GOSUB 5000
1050 ON BREAK GOSUB 1090
1060 GOSUB 2000
1070 GOSUB 3000
1080 IF INKEY$="" THEN GOTO 1080
1090 CLEAR:CLS:LIST 4000-4999
1100 END
1110 PRINT "POSIBLEMENTE EL FORMATO DE L
A SENTENCIA DATA NO ES CORRECTO"
1120 FOR i-1 TO 3000:NEXT i:GOTO 1090
```

Comentarios
Líneas 1030 y 1110-1120: Un breve mensaje de error para indicar que probablemente hay un error en la disposición del módulo DATA. No se trata de una prueba infalible, ya que algunas equivocaciones no generarán errores detectables por ON ERROR.

Comprobación
Cambie una de las cifras que aparecen bajo el encabezamiento «CANTIDAD POR ELEMENTO» por una letra, y lance el programa. Debería ver el mensaje de error del programa, y posteriormente se debería listar automáticamente el módulo DATA. Corrija el error deliberado y lance otra vez el programa. Esta vez, la pulsación por dos veces consecutivas de la tecla ESC (o la pulsación de cualquier tecla, una vez se haya terminado la gráfica) debería provocar el listado del módulo DATA.

PROGRAMA 2.3: Gráficos-3D

Función del programa

Habiendo examinado dos medios diferentes de presentar datos en el modo de alta resolución, resulta acertado recordar la enorme flexibilidad que proporciona el excelente conjunto de gráficos en baja resolución del 464. El uso de los caracteres gráficos de baja resolu-

ción que no están disponibles en el teclado, pero que puede ver en el Apéndice 3 del manual, ofrece al usuario la posibilidad de efectos confeccionados de antemano que serían extremadamente difíciles de conseguir en alta resolución.

En este programa crearemos un diagrama de barras tridimensional cuya presentación en pantalla es, creo, una de las mejores demostraciones de lo impresionante que puede llegar a ser el modo de baja resolución del 464 (en realidad el tipo de presentación visual que le hará llamar a la familia para mostrarles cuán inteligente es usted).

Los nuevos conceptos que se introducen a lo largo del programa son:

1) El uso del Datacorder (casete incorporado) para almacenar datos para el programa.
2) El uso de los caracteres gráficos de baja resolución.
3) Entrada interactiva de datos.

Módulo 2.3.1: Inicialización

Un módulo sencillo para declarar una pequeña matriz y para preguntar si se desea cargar datos desde la cinta. De todo ello se hablará posteriormente.

Módulo 2.3.1: Líneas 2000 - 2080

```
2000 REM*******************************
2010 REM Inicializacion
2020 REM*******************************
2030 CLS
2040 PRINT TAB(15);"GRAFICOS 3D";TAB(15)
;"===========":PRINT
2050 DIM hh(2,6)
2060 INPUT "Cargamos datos desde cinta (
s/n)";q$
2070 r$=CHR$(13)
2080 RETURN
```

Comentarios

Línea 2050: La matriz HH se utilizará para guardar los datos de la gráfica. Puesto que las cifras de la sentencia DIM deben contarse desde cero, quiere decirse que se ha reservado espacio para tres conjuntos de siete datos cada uno.

Línea 2060: Este INPUT está preparado para permitir que un módulo

posterior pueda llamar, desde cinta, a un conjunto de datos para una gráfica.
Línea 2070: La cadena R$ se utilizará en el módulo de fichero de datos, por lo que su explicación puede esperar hasta ese módulo.

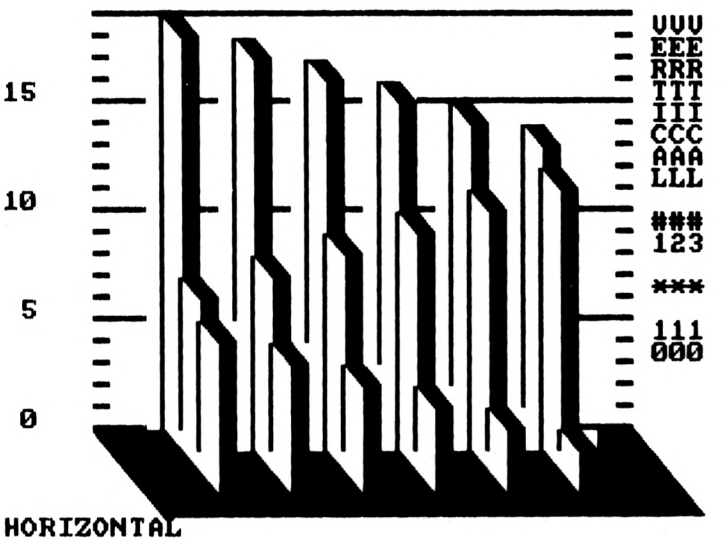


Fig. 2.3 Visualización del programa Gráficos-3D.

Módulo 2.3.2: Aceptación de datos

En el caso de este último programa del capítulo no adoptaremos la treta de usar sentencias data para almacenar información cambiante. La mayoría de los programas que trabajan con información útil, proporcionarán al usuario la posibilidad de introducir dicha información mientras el programa está en ejecución (a estos programas se les denomina «interactivos»). En este programa toda la información puede ser reunida de una vez, así que lo que vemos es un módulo que solicita información, que utiliza esa información para pedir más y que realiza algunas comprobaciones para detectar equivocaciones en las entradas.

Módulo 2.3.2: Líneas 3000 - 3250

```
3000 REM*******************************
3010 REM Aceptar datos
3020 REM*******************************
3030 CLS
```

```
3040 PRINT TAB(15);"GRAFICOS 3D";TAB(15)
;"===========":PRINT
3050 PRINT "Hay 19 divisiones en el eje
vertical.":PRINT
3060 INPUT "***Que cifra representara ca
da division:",uv:PRINT
3070 INPUT "Numero de columnas (1 a 6):"
,nd:PRINT
3080 INPUT "Numero de bancos (1 a 3):",n
b:PRINT
3090 PRINT "**************************
***********":PRINT
3100 INPUT "Nombre para el eje horizonta
l:",nh$:PRINT
3110 FOR i=0 TO nb-1
3120 PRINT "Nombre para el eje vertical"
;i+1;":";:INPUT nv$(i):PRINT
3130 NEXT i
3140 CLS
3150 FOR i=0 TO nb-1
3160 FOR j=1 TO nd
3170 t=20*uv
3180 WHILE t/uv>19
3190 PRINT:PRINT "Introducir banco";i+1;
"valor";j;":";
3200 INPUT t
3210 IF INT(t/uv)>19 THEN PRINT:PRINT"**
Valor demasiado grande**"
3220 WEND
3230 hh(i,j)=t
3240 NEXT j,i
3250 RETURN
```

Comentarios

Líneas 3050-3060: Como en el programa anterior del gráfico lineal, cada división del eje vertical puede representar cualquier valor especificado por el usuario. Observe que, por estar trabajando en baja resolución, no tenemos la misma flexibilidad que en el programa anterior en lo referente al tamaño de las divisiones verticales. El único tamaño práctico para cada división es la altura de un carácter, y el formato de la gráfica permitirá 19 divisiones en el eje vertical.

Líneas 3070-3080: Como se mencionó anteriormente, el programa permitirá que se presenten tres conjuntos de seis datos. Estos datos se presentarán en forma de tres líneas de hasta seis barras cada una. A cada línea de barras se le llamará un «banco», y a las barras en sí se les llamará «columnas».

Líneas 3110-3130: Puesto que puede haber hasta tres bancos, se pueden necesitar hasta tres nombres para el eje vertical (p. ej.: precio de venta, costo, beneficio). Observe que los nombres han de ponerse en la matriz NV$ (Nombres del eje Vertical) a partir del elemento 0; esta matriz no se dimensionó en el módulo de inicialización porque siempre tendrá únicamente tres elementos. La simple mención del nombre de una matriz en un programa la ajusta automáticamente a 10 elementos (0-9).
Líneas 3150-3240: Se le pide al usuario que introduzca, para cada banco, los datos correspondientes al número de columnas especificado. Se comprueba el número introducido para asegurar que no hará que el gráfico se salga por encima de la marca 19. Este es el propósito del bucle de las líneas 3180-3220: asegurar que no se introduzca ninguna cifra que requiera un gráfico superior a 19 caracteres en pantalla. Si se introdujese tal cifra, se avisa al usuario y se le pide que vuelva a introducir otra vez el dato.

Comprobación

Introduzca las líneas siguientes (finalmente algunas de ellas formarán parte del módulo de control del programa):

```
1030 GOSUB 2000
1040 GOSUB 3000
1110 CLS:END
```

Ahora ejecute lo que se ha introducido de programa y conteste a sus preguntas de la forma siguiente:
Para la cifra que representará cada división del eje vertical: 1
Columnas: 1
Bancos: 1
Nombre del eje horizontal: HORIZONTAL
Nombre del eje vertical 1: VERTICAL 1
Banco 1, valor 1: 10
Tras la entrada de la última cifra se borrará la pantalla y el programa se detendrá con el mensaje «READY» (PREPARADO...para nuevas órdenes). Ahora introduzca:

```
?UV,ND,NB,NH$,NV$(0),HH(0,1)
```

y el resultado debería ser:

```
1               1               1

HORIZONTAL      VERTICAL 1      10
```

Si lo desea, puede lanzar el programa otra vez e intentar introducir valores falsos. No debería ser posible introducir ningún valor de columna que sea mayor que 19 veces el valor de cada división del eje vertical.

Módulo 2.3.3: Trazado de la plantilla o estructura de la gráfica

Como el gráfico lineal, este otro también necesita que se trace una plantilla para que la información presentada tenga sentido. La tarea se lleva a cabo por este módulo, que utiliza bucles, variables y la orden LOCATE para ubicar los caracteres de baja resolución en la posición correcta de la pantalla.

Módulo 2.3.3: Líneas 4000 - 4260

```
4000 REM*********************************
4010 REM Trazado de la estructura
4020 REM*********************************
4030 CLS
4040 FOR i=0 TO 3
4050 LOCATE 6+i,21+i
4060 PEN 2:PRINT CHR$(213);STRING$(29,CH
R$(143));CHR$(215)
4070 NEXT i:PEN 1
4080 LOCATE 6,1:PRINT STRING$(30,CHR$(21
0))
4090 FOR i=2 TO 20
4100 LOCATE 6,i:PRINT CHR$(210)
4110 LOCATE 35,i:PRINT CHR$(210)
4120 NEXT i
4130 FOR i=5 TO 20 STEP 5
4140 LOCATE 6,i:PRINT STRING$(30,CHR$(21
0))
4150 NEXT i
4160 LOCATE 1,25:PRINT nh$
4170 FOR h=0 TO nb-1
4180 PEN h+1
4190 tt$=nv$(h)+" *"+STR$(uv)
4200 FOR i=1 TO LEN(tt$)
4210 LOCATE 37+h,i+1:PRINT MID$(tt$,i,1)
4220 NEXT i,h
4230 FOR i=0 TO 15 STEP 5
4240 LOCATE 1,20-i:PRINT USING "##";i
4250 NEXT i
4260 RETURN
```

Comentarios

Líneas 4040-4070: Este bucle dibuja una base sobre la que descansará la gráfica. Los caracteres gráficos están listados en el Apéndice 3 de su manual.

Línea 4080: Una línea a lo largo de la parte superior de la pantalla. Esta línea está formada por el carácter 210 (una barra horizontal) repetido 30 veces.

Líneas 4090-4120: Las dos tiras verticales de marcas, que se colocan a ambos lados de la gráfica para representar las 19 divisiones.

Líneas 4130-4150: Cuatro líneas horizontales, que atraviesan la gráfica de parte a parte, separadas entre sí por cinco divisiones del eje vertical.

Línea 4160: El rótulo del eje horizontal.

Líneas 4170-4220: Estos bucles presentan el nombre(s) del eje vertical (con cada carácter debajo del anterior), en el lado derecho de la pantalla. Se presenta también el valor que representa cada unidad vertical. La variable I de bucle se utiliza, tanto para ir de un nombre a otro del eje vertical, como para cambiar el color de cada nombre (color que se corresponde con el color del banco asociado).

Líneas 4230-4250: Este bucle pone números a la izquierda de la gráfica, próximos a las marcas de las divisiones múltiplos de 5.

Comprobación

Introduzca las líneas siguientes:

```
1050 GOSUB 4000
1070 WHILE INKEY$=""
1080 WEND
```

y la línea provisional

```
1085 PEN 5
```

y lance el programa. Especifique el valor correspondiente a cada división, una columna y tres bancos. Dé tres nombres para el eje vertical. Cuando se le pida que especifique el valor de las columnas, pulse simplemente RETURN. Debería ver visualizada entonces la plantilla de la gráfica, con el nombre del eje horizontal en la parte inferior y los nombres del eje vertical en la parte superior derecha.

Para continuar, pulse cualquier tecla y no olvide suprimir la línea 1085.

Módulo 2.3.4: Trazado de la gráfica

Se trata de un módulo que, francamente, es mucho más fácil de entender una vez que se ha introducido y se puede ver su efecto. El módulo no está concebido tanto en base a principios de carácter general, como en los resultados de experiencias llevadas a cabo para ver qué combinaciones de caracteres, ubicadas en qué posiciones, proporcionaban el efecto deseado.

Módulo 2.3.4: Líneas 5000 - 5220

```
5000 REM*********************************
5010 REM Trazado de las barras
5020 REM*********************************
5030 FOR h=0 TO nb-1
5040 PEN h+1
5050 FOR i=nd TO 1 STEP -1
5060 condicion=1:WHILE INT(hh(h,i)/uv)<>
0 AND condicion
5070 FOR j=1 TO INT(hh(h,i)/uv)+1
5080 LOCATE 9+4*(i-1)+h,22+h-j
5090 IF j=1 THEN PEN 2:PAPER 0:PRINT CHR
$(143);CHR$(215);:PEN h+1:PRINT CHR$(143
)
5100 IF j>1 AND j<INT(hh(h,i)/uv)+1 THEN
 PRINT CHR$(209);" ";CHR$(143)
5110 IF j=INT(hh(h,i)/uv)+1 THEN PRINT C
HR$(209);CHR$(213);CHR$(215)
5120 NEXT j
5130 condicion=0:WEND
5140 NEXT i
5150 NEXT h
5160 FOR i=0 TO nd-1
5170 FOR j=1 TO nb
5180 LOCATE 9+4*i+j,20+j
5190 IF j>1 OR hh(2,i)=0 OR (j=1 AND nd=
0) THEN PEN 2:PRINT CHR$(215)
5200 NEXT j
5210 NEXT i
5220 RETURN
```

Comentarios

Líneas 5030-5150: Este bucle crea el número de bancos especificado.

Líneas 5050-5140: El segundo bucle creará, trabajando de derecha a izquierda, el número de columnas especificado.

Líneas 5060-5130: Estas líneas crean un bucle WHILE ficticio, para

omitir la sección que traza una columna, si el valor de esa columna es 0 (el bucle es ficticio porque bajo ninguna circunstancia se ejecutará más de una vez). Esto se asegura haciendo del valor de la variable CONDICION una de las condiciones del bucle a repetir. Poniendo CONDICION a 0 antes de que se alcance WEND en la línea 5130, se asegura que el bucle no sea ejecutado nunca más de una vez. Se trata de una técnica útil, a recordar, ya que el 464 carece de una orden EXIT (orden de salida) que permitiría que el bucle finalizara automáticamente.

Líneas 5070-5120: Este bucle construye una columna. La línea 5080 da la posición para trazar cada carácter de la columna. La posición comenzará por la columa de la derecha (representada por I), subiendo por cada columna particular (dictado por J) y trasladándose cuatro espacios a la izquierda para cada columna nueva. Además, cuando se acaba un banco, el siguiente banco de columnas (representado por H) será presentado un espacio hacia abajo y hacia la derecha respecto al banco anterior.

La variable J de bucle está, como se mencionó, preparada para cambiar desde 1 hasta la altura de la columna. En la primera pasada del bucle se crea la base de la columna (línea 5090). En las pasadas posteriores, se utilizan diferentes caracteres para representar la parte lateral y frontal de la columna conforme se va construyendo, utilizando la orden LOCATE para ubicar la posición de presentación. Por último, se añade la parte superior de la columna (línea 5110).

Líneas 5160-5210: Cuando se han terminado las columnas quedan algunos bordes mal acabados en la parte inferior izquierda de cada una de ellas. Estas líneas se encargan de dejarlas bien terminadas.

Comprobación

Introduzca una línea nueva:

```
1060 GOSUB 5000
```

y lance el programa. Especifique 1 para el valor de las divisiones, tres columnas y tres bancos. Los nombres dados a los ejes carecen de importancia, así que haga su propia elección. Cuando se le pida el valor de cada columna, introduzca los siguientes:

```
6, 12, 18, 6, 12, 18, 6, 12, 18
```

debería ver los tres bancos y las tres columnas claramente presentados, con la parte superior de las tres columnas dando el aspecto de una superficie uniforme que va desde el banco trasero hasta el frontal. Observe que, en la lectura de los valores para los tres bancos, debe

suponer que la parte superior de la barra que hay más al frente continúa (en diagonal) hacia atrás y hacia arriba hasta la posición más atrasada, leyéndose el valor de la columna a partir de su borde trasero (como si apareciera en el banco que hay más atrás). En el ejemplo de la pantalla, lo que tiene son tres columnas, con las tres barras de cada columna representando el mismo valor, aunque la barra frontal está físicamente más baja en la pantalla. Esto es algo necesario para crear la ilusión tridimensional.

Experimente con el programa para ver cómo se ocupa de otros valores en los datos. Encontrará que sólo funcionará realmente bien siempre que un banco no sea más alto que el que hay detrás de él. Hay una gran cantidad de datos que se ajustan a este tipo de disposición tales como los datos precio de venta/costo/beneficio mencionados anteriormente.

Módulos 2.3.5 y 2.3.6: Almacenamiento de datos en cinta

Pasamos ahora, por primera vez, a un tema que muchas personas dejan a sus expensas en sus programas: el almacenamiento de datos en cinta. Sin duda alguna, ya habrá observado que cuando ha cometido uno o dos errores en el tecleado del programa, la reintroducción de los datos (al poco tiempo de haberlos introducido) puede convertirse en algo un poco más que molesto. Además, en el caso de muchos programas, tiene poco sentido poner los datos en el ordenador, si de todas formas tiene que volver a acordarse de ellos cada vez que desconecta el 464; quizás, en el caso de este programa en concreto, no sería demasiado impracticable introducir los datos de nuevo, pero ¿qué me diría sobre programas más complejos en los que puede haber literalmente cientos de unidades de información?

Todos estos problemas se pueden superar mediante el uso constante de la platina (Datacorder), para grabar los datos que se han introducido y, habiéndolos grabado, para reintroducirlos en el 464 siempre que el usuario lo desee. Esa tarea la llevan a cabo los dos módulos que se presentan a continuación.

Módulos 2.3.5 y 2.3.6: Líneas 6000 - 7110

```
6000 REM*******************************
6010 REM Guardar datos en cinta
6020 REM*******************************
6030 OPENOUT "datos-grafico"
6040 PRINT #9,nb;r$;nd;r$,nh$;r$;uv
6050 FOR i=0 TO nb-1
6060 PRINT #9,nv$(i)
```

```
6070 FOR j=0 TO nd
6080 PRINT #9,hh(i,j)
6090 NEXT j,i
6100 PRINT #9:CLOSEOUT
6110 RETURN
7000 REM*******************************
7010 REM Cargar datos desde la cinta
7020 REM*******************************
7030 OPENIN "datos-grafico"
7040 INPUT #9,nb,nd,nh$,uv
7050 FOR i=0 TO nb-1
7060 INPUT #9,nv$(i)
7070 FOR j=0 TO nd
7080 INPUT #9,hh(i,j)
7090 NEXT j,i
7100 CLOSEIN
7110 RETURN
```

Comentarios

Línea 6030: Antes de poder almacenar los datos en cinta, es necesario preparar un sitio para ellos, proceso al que se le conoce como «apertura de un fichero». El formato para esto es:

OPENOUT «NOMBRE-DE-FICHERO»

Líneas 6040-6090: Algo a observar en la sentencia PRINT # es la presencia de varios R$ en la línea. Es posible que recuerde que, en el primer módulo del programa, R$ se hizo igual a CHR$(13), que no es otra cosa que el carácter RETURN (retorno de carro) cuya presencia indica el final de un dato a presentar. Cuando se cargan varios datos en un fichero, mediante una única sentencia PRINT #, si no se incluye un R$ entre los datos, todos ellos serán lanzados juntos (sin separación alguna).

Líneas 6060 y 6080: En el comentario sobre la última línea se dijo que hay que incluir R$ (o cualquier otra variable igual a CHR$(13)) para separar los datos. Entonces, ¿por qué no se hace igual en el caso de estos dos bucles que meten el contenido de las dos matrices en el fichero de cinta? La respuesta es que, siempre que una sentencia PRINT o PRINT # termina sin signo alguno de puntuación, el 464 pone automáticamente el carácter RETURN detrás del último dato introducido (esa es la razón por la que los datos se presentan en líneas independientes de la pantalla cuando los datos anteriores no tienen ninguna coma o ningún punto y coma tras ellos).

Línea 6100: Cuando se están metiendo datos en un fichero es prudente acabar con un sencillo PRINT #<NUMERO-DE-FICHERO>

vacío, esto asegura que se borrará cualquier dato que esté esperando aún en la memoria del 464, para entrar en el fichero. Por último, el fichero debe ser cerrado (CLOSE), utilizándose CLOSEOUT para un fichero en el que se han salvado (grabado) datos. No hacer esto, significará que el número de fichero no estará disponible para usos futuros, lo que puede llevar incluso a la pérdida de los datos de la cinta cuando el programa intente releerlos.
Líneas 7000-7110: Estas líneas llevan a cabo la operación opuesta a la descrita anteriormente; es decir, lo que hacen es llamar a los datos almacenados previamente en la cinta.
Línea 7030: Una vez más se debe abrir un fichero (OPEN). La única diferencia con la sentencia OPEN anterior es que ahora utilizamos la forma OPENIN de OPEN, lo que le dice al sistema que lo que queremos es coger datos *desde* la cinta.
Líneas 7040-7090: Lo opuesto de PRINT # es INPUT #. Observe que no tenemos que hacer uso del separador R$ cuando se están metiendo (INPUT) datos. En la propia naturaleza de INPUT e INPUT # está el que no reconozcan que han recibido un dato hasta que se pulsa un ENTER o se lee un carácter RETURN desde la cinta.
Línea 7100: Al igual que en la línea 6100, el fichero *debe* cerrarse cuando se ha terminado con él.

Comprobación

Es mejor dejar la comprobación de este módulo hasta que haya introducido las pocas líneas que serán necesarias para completar el módulo de control del programa.

Módulo 2.3.7: El módulo de control

Al ir haciendo los procedimientos de comprobación de los módulos anteriores ha introducido la mayoría de las líneas de este módulo. Todo lo que resta es asegurarse de que el módulo está completo, comparándolo con el listado que se da más abajo.

Módulo 2.3.7: Líneas 1000 - 1110

```
1000 REM*******************************
1010 REM Control
1020 REM*******************************
1030 GOSUB 2000
1040 IF LOWER$(q$)="s" THEN GOSUB 7000 E
LSE GOSUB 3000
1050 GOSUB 4000
1060 GOSUB 5000
```

```
1070 WHILE INKEY$=""
1080 WEND
1090 LOCATE 1,25:INPUT "Guardamos datos
en cinta (s/n)";q$
1100 IF LOWER$(q$)="s" THEN GOSUB 6000
1110 CLS:END
```

Comprobación

Simplemente lance el programa. Debería poder ahora introducir los datos para crear un gráfico. Cuando el gráfico esté en pantalla, la pulsación de cualquier tecla debería provocar la petición (por parte del programa) de grabado en cinta de los datos. Antes de responder «S» asegúrese de que hay una cinta adecuada en el Datacorder (no creo que desee regrabar encima de algún programa interesante). Cuando el programa termine la grabación de sus datos, láncelo otra vez y responda «S» cuando se le pregunte si desea cargar desde la cinta. Realizada la operación de carga, debería volver a ver el mismo gráfico en pantalla. Si la prueba es satisfactoria, el programa está listo para ser usado.

3. Son et lumière

En este capítulo miraremos más de cerca las aptitudes del 464 en los campos del sonido y los gráficos. En la mayoría de los casos, usted escribirá programas que tendrán pequeñas rutinas incorporadas para realizar tareas gráficas o sonoras, incluso aún cuando sólo sea algo tan trivial como el sencillo módulo de título ilustrado en el último capítulo, o la señal de alarma de dos tonos. Sin embargo, en otros casos se necesitan cosas más ambiciosas: un diseño complejo o una pieza musical para alegrar un programa. En tales casos, en vez de escribir cada vez que lo necesite una rutina independiente para crear el diseño o la melodía, es más práctico tener algunos instrumentos a mano que le permitan crear diseños más fácilmente, y a los que pueda llamar desde la cinta para usarlos en programas posteriores.

En este capítulo encontrará tres de tales instrumentos que le permitirán crear diseños en alta resolución, crear un conjunto propio de caracteres y escribir y editar música en su 464. Con los medios que le proporcionarán los programas de este capítulo, el único límite para usar sonidos y gráficos en sus programas será la imaginación que ponga en juego.

Los programas que se incluyen en este capítulo son:

CARACTERES: Permite crear juegos de caracteres a gusto del usuario.

DISEÑADOR: Un instrumento para crear diseños en alta resolución.

MUSICA: Le permite introducir melodías a tres voces en un formato sencillo, así como su audición posterior.

PROGRAMA 3.1: Caracteres

Función del programa

Habiendo visto en el capítulo anterior parte de lo que se puede conseguir con los gráficos en alta resolución nos marchamos al otro

lado del espectro: los gráficos en baja resolución, con un programa que le permite cambiar la forma de los caracteres que el 464 presenta en la pantalla. Pero antes de pasar al programa propiamente dicho, es necesario dar una pequeña explicación sobre la forma en que se crean y presentan los caracteres en el modo de baja resolución.

La pantalla de baja resolución del 464 tiene sitio para 25 líneas, cada una de ellas de 40 caracteres, lo que da un total de 1000 posiciones de carácter. En otras palabras, puede presentar 1000 elementos distintos en pantalla aunque algunos de ellos tendrán que ser iguales, ya que el 464 no puede generar 1000 caracteres *diferentes* al mismo tiempo. Pero ese mil no es todo lo que nos interesa; si mira de cerca cualquier carácter de la pantalla observará que no está trazado con una línea continua, como las palabras que está leyendo ahora, sino con puntos. En realidad, cada una de las mil posiciones de carácter de la pantalla está compuesta por 64 puntos, y son las combinaciones de esos 64 puntos las que constituyen cada uno de los caracteres que puede visualizar el 464. La letra «A», por ejemplo, está trazada según se muestra en la figura 3.1.

```
                ====================
FILA 1 -->          0 0
     2 -->        0 0 0 0
     3 -->      0 0       0 0
     4 -->      0 0 0 0 0 0 0
     5 -->      0 0       0 0
     6 -->      0 0       0 0
     7 -->      0 0       0 0
     8 -->
                ====================
```

Fig. 3.1 Ampliación de la letra «A» según aparece en pantalla.

Los puntos a partir de los que se confeccionan los caracteres se denominan «pixels», lo que no es más que una contracción de «picture elements» (elementos de imagen), y representan el área más pequeña que el 464 puede manejar en pantalla, tanto en los modos de alta como de baja resolución.

Estas figuras complejas no aparecen por casualidad. Es evidente que en algún sitio de la memoria del 464 debe estar establecido que cuando pulsa la tecla etiquetada con una «A», debe aparecer en pantalla la configuración de puntos mostrada en la ilustración. En realidad, todos los caracteres que puede presentar el 464 están almacenados, en forma de números, en un bloque de memoria, al que se le conoce como «memoria de caracteres» o «ROM de caracteres». A cada carácter se le asignan ocho octetos de memoria, y cada uno de esos

octetos determina dónde aparecerán los pixels de cada una de las filas del carácter. Esto se hace convirtiendo el valor de cada octeto en una representación de la fila según el sistema de numeración binario utilizado por el 464, sistema en el que los números se expresan en términos de potencias de 2 y no de 10 como ocurre en nuestro sistema normal (decimal) de contar. Por ejemplo:

2013006

en nuestro sistema usual de contar, significa:

(2*10^6)+(0*10^5)+(1*10^4)+(3*10^3)+(0*10^2)+(0*10^1)+(6*10^0)

Sin embargo, en el sistema binario los únicos dígitos permitidos son «1» y «0», y un número tal como el:

11001010

significa:

(2^7)+(2^6)+(2^3)+(2^1)

donde se han ignorado los ceros.

No es necesario tener un conocimiento profundo del sistema binario, todo lo que necesita es recordar que un octeto de memoria del 464 almacena un número binario de ocho dígitos, y que todos esos unos y ceros son una forma perfecta de registrar qué pixels de una fila del carácter están activados y cuáles no. La letra «A», por ejemplo, está formada por los siguientes ocho números binarios (octetos):

0 0 0 1 1 0 0 0
0 0 1 1 1 1 0 0
0 1 1 0 0 1 1 0
0 1 1 0 0 1 1 0
0 1 1 1 1 1 1 0
0 1 1 0 0 1 1 0
0 1 1 0 0 1 1 0
0 0 0 0 0 0 0 0

Si mira atentamente aún puede ver la «A» bastante claramente, aunque trazada ahora con unos en vez de con pixels. Sin embargo, le costaría mucho tiempo reconocerla en los números:

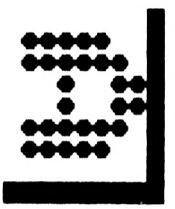
24, 60, 102, 102, 126, 102, 102, 0

que no es más que lo que son esos números binarios, cuando se escriben en nuestro más cómodo sistema de numeración decimal.

El objetivo de todo esto es llegar al hecho de que, cuando se pide que se presente un cierto carácter en pantalla, el 464 lo busca en la «memoria de caracteres» y utiliza lo que encuentra allí para trazarlo sobre la pantalla.

De todo esto se sigue que si fuera posible *cambiar* el contenido de la memoria de caracteres, cambiaría igualmente la forma de los caracteres presentados en pantalla. De esta forma, podríamos tener letras a nuestro gusto, nuevos caracteres gráficos o cualquier cosa que encaje dentro de la celdilla básica (8*8) de carácter.

El problema, sin embargo, es que la memoria de caracteres *no puede* ser alterada; forma parte de la ROM, la «memoria de sólo lectura» (Read Only Memory), incorporada de forma permanente al 464. Lo que podemos hacer es copiarla en algún lugar de la RAM, el tipo de memoria que sí puede ser cambiada, y utilizar entonces los datos copiados, junto con la potente orden SYMBOL, para modificar los caracteres. De esta forma, podemos juguetear tanto como queramos con el conjunto de caracteres, cosa que es el objetivo del programa que viene a continuación.



```
Numero del caracter: 65

'i' para invertir
'e' para imagen especular
'v' para volver
'1' para marcar casilla
'0' para borrar casilla
'r' para rotar
'p' para poner en memoria
'g' para grabar en cinta
'c' para cargar desde cinta
'x' para normalizar los caracteres
't' para terminar
Teclas de cursor para mover
```

Fig. 3.2 Visualización del programa Caracteres. En la figura se muestra la pantalla durante el modo de edición de caracteres, con una letra «A» a la que se ha rotado 90 grados.

Módulo 3.1.1: El módulo de control

Normalmente no metemos un módulo de control al principio del proceso de introducir un programa, pero éste en concreto es corto y sin él, el programa necesitará que se introduzca un GOSUB antes de que pueda arrancarse correctamente.

Módulo 3.1.1: Líneas 10000 - 10050

```
10000 REM*******************************
10010 REM Control
10020 REM*******************************
10030 GOSUB 11000
10040 GOSUB 12000
10050 CLS:END
```

Módulo 3.1.2: Cambio al nuevo conjunto de caracteres

El objetivo de este módulo es llevar a cabo la tarea esbozada en la introducción del programa: la de transferir los datos de los caracteres a alguna parte de la memoria en donde se los pueda manipular. En realidad, el sitio donde se almacenarán es una matriz que denominaremos C%, cosa que nos proporciona una gran flexibilidad en el almacenamiento y cambio de los datos.

Módulo 3.1.2: Líneas 11000 - 11190

```
11000 REM*******************************
11010 REM Inicializacion
11020 REM*******************************
11030 CLS:PEN 1:PRINT "Primer caracter a
 definir por usuario   ":INPUT "(32-255)
";pc
11040 IF pc<32 OR pc>255 THEN PRINT "**F
UERA DEL INTERVALO: PRUEBE OTRA VEZ**":G
OTO 11030
11050 SYMBOL AFTER pc
11060 IF in=0 THEN ca=pc:DIM a%(7),t%(7)
,c%(255,7):in=1
11070 menu$="iev10rpgcxt"+CHR$(240)+CHR$
(241)+CHR$(242)+CHR$(243)
11080 MODE 2
11082 LOCATE 36,14:PRINT "ESPERE POR FAV
OR"
```

```
11084 LOCATE 1,1
11090 FOR i=pc TO 255
11100 PRINT CHR$(i);
11110 NEXT i
11120 FOR i=0 TO 255-pc
11130 FOR j=0 TO 7
11140 c%(pc+i,j)=PEEK(48*1024+j*2048+i)
11150 NEXT j
11160 NEXT i
11170 MODE 1
11180 a$="v"
11190 RETURN
```

Comentarios

Líneas 11030-11050: El 464 le proporciona la posibilidad de definir desde 16 hasta sus 255 caracteres, aunque sólo puede definir más de 16 caracteres, si le dice explícitamente que eso es lo que desea, mediante la orden SYMBOL AFTER. Esta orden seguida de un número, significa que de ahora en adelante desea poder alterar, mediante la orden SYMBOL que veremos inmediatamente, cualquier carácter desde el representado por el número que sigue a SYMBOL AFTER hasta el número de carácter 255. Estas líneas le permiten fijar el primer carácter susceptible de ser redefinido (cualquiera desde el 32 hasta el 255). En uso normal, el 464 le permite redefinir 16 caracteres, los que van del 240 en adelante.

Línea 11060: Aparte de un caso específico cuando está siendo utilizado el programa, esta línea se utilizará para preparar la matriz principal (C%), que guardará los datos de los caracteres (256 elementos con 8 números en cada elemento), y dos matrices que son utilizadas para la manipulación transitoria de un carácter individual.

Línea 11070: Se trata de una línea importante, aunque su significado será explicado cuando pasemos al módulo que se ocupa del menú del programa.

Líneas 11080-11170: Un toque astuto. En vez de coger los datos de la memoria, es más sencillo tomarlos de un sitio obvio: la propia pantalla. La ventaja de ello es que, en un futuro, cuando desee coger los datos de ciertos caracteres seleccionados, todo lo que se necesitará será presentarlos en pantalla. Para hacer esto, primero cambiamos al MODO 2 de pantalla, el modo de alta resolución; la razón de ello es que en el MODO 2 la distribución de los pixels en pantalla se corresponde exactamente con la configuración descrita en la introducción del programa. En el MODO 1, los dígitos binarios individuales no son copiados exactamente en pantalla por los pixels, ya que se presentan dos pixels horizontalmente por cada dígito binario.

Habiendo cambiado el modo, la tarea siguiente es presentar todos los caracteres desde PC (el Primer Carácter especificado por el usuario) hasta 255, y desde la primera posición de pantalla en adelante. El segundo par de bucles (líneas 11120-11160) lee ahora los octetos de memoria que componen la presentación en pantalla. Esto no es algo tan sencillo como podría ser simplemente debido a la forma en que está dispuesta la memoria de pantalla. El bloque de memoria que guarda el contenido de la pantalla está dispuesto de tal forma (no entraremos en las razones) que si desea encontrar los ocho octetos que componen el carácter situado en la esquina superior izquierda (en MODO 2), entonces tendrá que mirar al octeto de memoria 49152 para obtener el octeto superior del carácter, después al 51200 (49152+2048) para conseguir el que está más abajo (el segundo de los que componen el carácter) y así sucesivamente, en saltos de 2048 octetos para cada uno de los ocho octetos que componen el carácter. El siguiente carácter a la derecha comienza por el octeto 49153 y continúa hacia abajo en saltos de 2048 octetos. Cuando se llega al final de la primera línea de caracteres, y nos movemos un octeto más, obtenemos el primer octeto del primer carácter de la segunda línea. Si con todo, no le ha quedado clara en absoluto la explicación anterior, intente meter el breve programa de prueba que se expone a continuación:

```
10 MODE 2
20 FOR I=49152 TO 49152+1024
30 POKE I,255
40 NEXT I
```

Verá que, aún cuando está activando (introduciendo 255, o lo que es lo mismo 11111111, en cada posición de memoria mediante POKE) la memoria de pantalla en saltos de un octeto, la línea generada por la activación de los pixels (su puesta a 1), está situada en la parte superior de las líneas de caracteres.

Con todo esto en su mochila, debería ser capaz de ver lo que hace la línea 11140 conforme va cogiendo (de la memoria de pantalla) octetos, uno a uno, de los datos del carácter.

Línea 11180: Esta línea prepara una variable que se utiliza para almacenar las respuestas al menú, de forma que el programa no finalice accidentalmente en la primera pasada.

Módulo 3.1.3: Presentación de un carácter ampliado

La esencia de este programa es que hará fácil la edición de caracteres. Una forma para conseguir esto es presentar una versión ampliada del carácter especificado, junto a un cursor que pueda moverse en torno a él. Este módulo posibilita la especificación del carácter y su posterior presentación ampliada en pantalla.

Módulo 3.1.3: Líneas 12000 - 12220

```
12000 REM*******************************
12010 REM Mostrar caracter ampliado
12020 REM*******************************
12030 WHILE a$<>"t"
12040 CLS:PEN 1
12050 FOR i=0 TO 7
12060 IF hecho=0 THEN a%(i)=c%(ca,i)
12070 b$=RIGHT$(BIN$(256+a%(i)),8)
12080 FOR j=1 TO 8
12090 IF MID$(b$,j,1)="1" THEN PRINT CHR
$(231); ELSE PRINT " ";
12100 NEXT j
12110 PEN 3:PRINT CHR$(143):PEN 1
12120 NEXT i:hecho=1
12130 PEN 3:PRINT STRING$(9,CHR$(143)):P
EN 1
12140 LOCATE 25,5:PRINT CHR$(ca)
12150 PEN 3:LOCATE 1,11
12160 PRINT "Numero del caracter:";ca:PR
INT
12170 IF a$="v" THEN INPUT "Numero para
mover puntero (0 para redefinir)";mm:ca=
ca+mm
12180 IF ca<pc THEN ca=pc
12190 IF ca>255 THEN ca=255
12200 IF mm=0 THEN GOSUB 13000 ELSE hech
o=0
12210 WEND
12220 RETURN
```

Comentarios

Líneas 12040-12120: Se transfieren los datos de carácter, para un carácter particular, desde C% a A% y entonces se utilizan para presentar en pantalla una versión ampliada del carácter. El medio para hacer esto, es utilizar primero BIN$ para transformar cada octeto de los datos del carácter en una cadena de unos y ceros; después se

visualiza un carácter «círculo» (CHR$(231)) en los sitios correspondientes a los unos de la cadena. La celda de 8*8 se rodea con un borde continuo construido a partir de CHR$(143) y, por último, se pone a 1 la variable HECHO, para que si el módulo se repite sin cambios en el carácter no se vuelvan a copiar los datos en A%.
Línea 12140: A la derecha de la versión ampliada se coloca una versión del carácter a tamaño normal.
Líneas 12170-12190: Cuando el programa se ejecuta por primera vez, el carácter mostrado será el primero (según el valor almacenado en la variable PC) de los que se pueden redefinir. Estas líneas hacen posible mover el carácter presentado en pantalla dentro del intervalo limitado por PC y 255.

Comprobación

Lance el programa y especifique el número 65 como el primero de los caracteres a definir por el usuario. Después de que se hayan visualizado todos los caracteres y de una pausa para la transferencia, debería ver una versión ampliada de la letra «A», trazada en la esquina superior izquierda de la pantalla. También debería poder moverse por el conjunto de caracteres, examinando cualquier carácter que desee desde la «A» en adelante. Observe que aún no puede *hacer* nada con el carácter visualizado; llegado el momento, esto se conseguirá pulsando un «0» para llamar a un módulo posterior. Normalmente el programa debería detenerse al introducirse el modo «0», pero en este módulo puede hacerlo pulsando ESC.

Módulo 3.1.4: Un cursor definido por el usuario

Un módulo sencillo para presentar un cursor «de imitación», en una posición seleccionada.

Módulo 3.1.4: Líneas 14000 - 14110

```
14000 REM*******************************
14010 REM Mostrar cursor
14020 REM*******************************
14025 b$=RIGHT$(BIN$(256+a%(y)),8)
14030 LOCATE x+1,y+1:PAPER 2:PEN 1
14040 IF MID$(b$,x+1,1)="1" THEN PRINT C
HR$(231); ELSE PRINT " ";
14050 a$=""
14060 WHILE a$=""
14070 a$=LOWER$(INKEY$)
14080 WEND
```

```
14090 LOCATE x+1,y+1:PAPER 0
14100 IF MID$(b$,x+1,1)="1" THEN PRINT C
HR$(231); ELSE PRINT " ";
14110 RETURN
```

Comentarios
Líneas 14030-14040: La apariencia del cursor se crea cambiando los colores de papel (PAPER) y tinta (INK), simulando la presencia del cursor normal. Se imprime entonces un espacio en blanco o el pequeño «círculo» utilizado para visualizar las versiones ampliadas de los caracteres.
Líneas 14050-14080: Un estado de espera hasta que se pulsa una tecla.
Líneas 14090-14100: Se reimprime con los colores normales la posición de carácter donde está localizado el cursor.

Comprobación
Introduzca

RUN 14000[RETURN]

y debería ver un cursor cuadrado en la esquina superior izquierda de la pantalla. Pulse cualquier tecla y el programa se detendrá con un mensaje de error «Unexpected RETURN».

Módulo 3.1.5: Introducción de órdenes

Este módulo combina la presentación de un menú, el movimiento del cursor parpadeante, el borrado o marcado de los pixels ampliados, y acepta órdenes para manipular el carácter de pantalla. Las instrucciones completas para su uso, se encuentran en el propio menú. El único aspecto original de este módulo es la forma en que son aceptadas las órdenes.

Módulo 3.1.5: Líneas 13000 - 13220

```
13000 REM******************************
13010 REM Redefinir caracter
13020 REM******************************
13030 LOCATE 1,13:PRINT STRING$(40," ")
13040 LOCATE 1,13
13050 PRINT "'i' para invertir"
13060 PRINT "'e' para imagen especular"
```

```
13070 PRINT "'v' para volver"
13080 PRINT "'1' para marcar casilla"
13090 PRINT "'0' para borrar casilla"
13100 PRINT "'r' para rotar"
13110 PRINT "'p' para poner en memoria"
13120 PRINT "'g' para grabar en cinta"
13130 PRINT "'c' para cargar desde cinta
"
13140 PRINT "'x' para normalizar el jueg
o de carac-      teres"
13150 PRINT "'t' para terminar"
13160 PRINT "teclas de cursor para mover
"
13170 ret=0:WHILE ret=0
13180 GOSUB 14000
13190 z=INSTR(menu$,a$)
13200 ON z GOSUB 15000,16000,17000,18000
,19000,20000,21000,22000,23000,24000,250
00,26000,27000,28000,29000
13210 WEND
13220 RETURN
```

Comentarios

Líneas 13170-13210: Quizá recordará que en el módulo de inicialización preparamos una cadena bastante poco usual, denominada MENU$. Estas líneas son la razón; todas ellas juntas permiten que una única pulsación de tecla sea transformada económicamente, mediante INSTR, en un número. La posición de un carácter dentro de MENU$, es el valor que se le dará a Z, y el valor utilizado en el ON Z GOSUB de la línea siguiente. La variable RET se utiliza para indicar al módulo si necesita retrazar la presentación completa. Si RET es igual a 0 cuando retorna de la ejecución, entonces el carácter ampliado no es retrazado, con el consiguiente ahorro de tiempo.

Comprobación

Ejecute el programa. Cuando se haya trazado la letra «A», debería poder meter un «0» y ver el menú en pantalla. Ninguna de las órdenes tendrá ningún efecto, excepto el de detener el programa con un mensaje de error.

Módulo 3.1.6: Volviendo del modo de edición

Pulsando «V» en el módulo de menú la ejecución vuelve al módulo 3.1.3, permitiendo al usuario moverse a otro carácter después de haberse ocupado de uno.

Módulo 3.1.6: Líneas 17000 - 17030

```
17000 REM*******************************
17010 REM Vuelta (retorno)
17020 REM*******************************
17030 ret=1:RETURN
```

Comprobación

Ahora debería ser capaz de alternar entre los módulos de menú principal y de movimiento de carácter.

Módulo 3.1.7: Movimientos del cursor

Habiéndonos proporcionado un cursor, añadimos ahora la capacidad de moverlo, lo que se hace alterando simplemente los valores de las variables X e Y de las coordenadas del cursor.

Módulo 3.1.7: Líneas 26000 - 29040

```
26000 REM*******************************
26010 REM Cursor hacia arriba
26020 REM*******************************
26030 IF y>0 THEN y=y-1
26040 RETURN
27000 REM*******************************
27010 REM Cursor hacia abajo
27020 REM*******************************
27030 IF y<7 THEN y=y+1
27040 RETURN
28000 REM*******************************
28010 REM Cursor a la izquierda
28020 REM*******************************
28030 IF x>0 THEN x=x-1
28040 RETURN
29000 REM*******************************
29010 REM Cursor a la derecha
29020 REM*******************************
29030 IF x<7 THEN x=x+1
29040 RETURN
```

Comprobación

Con el menú principal en pantalla tendría que ser posible mover el cursor en torno al carácter ampliado; sin embargo, aún no puede hacer ningún tipo de alteración en el diseño.

Módulo 3.1.8: Entintando y borrando

Pulsando «1» o «0» desde el menú principal se marca (entinta) o se borra un pixel individual del diseño ampliado. Esta tarea es realizada por este módulo, con el concurso de los operadores lógicos AND y OR.

Módulo 3.1.8: Líneas 18000 - 19060

```
18000 REM*******************************
18010 REM Marcar casilla
18020 REM*******************************
18030 a%(y)=a%(y) OR 2^(7-x)
18040 LOCATE x+1,y+1
18050 PRINT CHR$(231)
18060 RETURN
19000 REM*******************************
19010 REM Borrar casilla
19020 REM*******************************
19030 a%(y)=a%(y) AND 255-2^(7-x)
19040 LOCATE x+1,y+1
19050 PRINT " "
19060 RETURN
```

Comentarios

Línea 18030: El uso del operador OR es un método común de cambiar a «1» o más de los dígitos binarios de un número. Lo que hace OR es comparar dos números binarios y generar entonces un tercero en el que cada dígito binario es 1 si el dígito binario de *cualquiera* de los números originales era 1. Por ejemplo, si los dos números originales fueran:

124 = 01111100

y

3 = 00000011

el resultado sería 127, o 01111111, puesto que cada dígito binario era 1 (excepto el más significativo) en uno u otro número.

Si queremos asegurarnos de que un dígito binario particular sea puesto a 1 (digamos el que hace el número N contando de derecha a izquierda), todo lo que debemos hacer es utilizar OR 2^N.

En el caso del programa activar un dígito binario particular es lo mismo que activar un pixel, puesto que el estado de los pixels está determinado por el estado de los dígitos binarios de los ocho números que componen los datos del carácter.

Línea 19030: El operador opuesto al OR, en lo referente a la activación/desactivación de dígitos binarios individuales, es el operador AND. Cuando dos números se carean mediante un AND, el número resultante sólo tiene dígitos puestos a 1 en aquellas posiciones en que *ambos* números originales los tuvieran. Así, por ejemplo, un AND entre 124 y 3 daría como resultado 0, ya que ninguno de los dígitos binarios está activado en ambos números. Un AND entre un cierto número, comprendido entre 0 y 255, y 255 no altera al número en absoluto, ya que en el número 255 cada uno de los ocho dígitos está puesto a 1. Sin embargo, se puede desactivar cualquiera de los dígitos de 255, restando 2^N de 255 (donde N es el número del dígito que se desea poner a 0). En consecuencia, un AND entre un cierto número, digamos X, y 255-2^N, pone a 0 al bit N de X.

En términos del programa esto es equivalente a borrar un pixel.

Comprobación

En el modo de edición de caracteres, ahora debería poder mover el cursor de un lado a otro, marcando o borrando pixels a voluntad.

Módulo 3.1.9: Creación de un carácter inverso

Empezamos ahora una serie de tres módulos que haciendo operaciones sobre todo el carácter, operaciones como transformarlo en su imagen especular o girarlo 90 grados, hacen un poco más fácil la tarea de edición de dicho carácter. Este módulo invierte (los colores de fondo y texto) el carácter de pantalla. Cualquier pixel que estuviera activado será borrado, y a cualquier posición que estuviera vacía se le pondrá un pixel activado. Esto se consigue invirtiendo los 1 y 0 binarios de los octetos que componen el carácter.

Módulo 3.1.9: Líneas 15000 - 15060

```
15000 REM*******************************
15010 REM Invertir caracter
15020 REM*******************************
15030 FOR i=0 TO 7
15040 a%(i)=255-a%(i)
```

```
15050 NEXT i
15060 ret=1:RETURN
```

Comprobación

Lance el programa y meta el modo de edición. Cuando el menú esté en pantalla pulse «I» y verá crearse el carácter invertido. Pulse «I» otra vez y el carácter volverá a su apariencia normal. Observe que todo esto sólo se refiere al carácter ampliado, no al de tamaño normal situado a la derecha de la celda de 8*8. El carácter de tamaño normal sólo cambiará si decide introducir en memoria su carácter editado, cosa que no se podrá hacer hasta que no se introduzca un módulo posterior.

Módulo 3.1.10: Creación de una imagen especular

Este módulo toma el carácter presentado en pantalla y «le da la vuelta», como si se estuviera mirando en un espejo.

Módulo 3.1.10: Líneas 16000 - 16100

```
16000 REM*******************************
16010 REM Imagen especular
16020 REM*******************************
16030 FOR i=0 TO 7
16040 b$=RIGHT$(BIN$(256+a%(i)),8)
16050 a%(i)=0
16060 FOR j=0 TO 7
16070 a%(i)=a%(i)+2^j*VAL(MID$(b$,j+1,1)
)
16080 NEXT j
16090 NEXT i
16100 ret=1:RETURN
```

Comentarios

Líneas 16060-16080: Conforme se extrae cada octeto de A% y se transforma en una cadena binaria, se va leyendo de izquierda a derecha. Así, cuando el valor de la variable de bucle J es 0, el dígito que se está leyendo en ese momento representa 128. De forma que si el primer carácter de la cadena binaria es un 1, entonces es transformado en 2^0 (1 en términos decimales). Si el último carácter de la cadena es un 1, entonces es transformado en 2^7 (128 en términos decimales). De esta forma se intercambian los dígitos del número binario que ocupan posiciones simétricas «en el sentido izquierda/derecha».

Comprobación

Lance el programa y llame al menú de edición. Pulse «E» y tras una pausa, durante la cual el carácter está siendo copiado en la matriz, debería verlo presentado en forma especular.

Módulo 3.1.11: Rotación de un carácter

Si piensa en el carácter que está editando como si estuviera impreso sobre una hoja de plástico transparente, entonces, aparte de sostenerlo a un cierto ángulo, todo lo que pudiera hacer con la hoja de plástico puede realizarlo mediante una combinación de imágenes especulares y/o una o más rotaciones de 90 grados. El módulo que nos ocupa utiliza otra vez la matriz R%, pero ahora para girar el carácter 90 grados (en sentido horario) dentro de la celda de 8*8.

Módulo 3.1.11: Líneas 20000 - 20130

```
20000 REM*******************************
20010 REM Rotar 90 grados
20020 REM*******************************
20030 FOR i=0 TO 7
20040 b$=RIGHT$(BIN$(256+a%(i)),8)
20050 FOR j=0 TO 7
20060 IF i=0 THEN r%(j)=0
20070 r%(j)=r%(j)+2^i*VAL(MID$(b$,j+1,1)
)
20080 NEXT j
20090 NEXT i
20100 FOR i=0 TO 7
20110 a%(i)=r%(i)
20120 NEXT i
20130 ret=1:RETURN
```

Comentarios

Líneas 20050-20090: Este bucle copia los dígitos binarios, de cada uno de los ocho números de A%, en la matriz provisional R%. Observe, sin embargo, que mientras los dígitos binarios de cada número de A% se están leyendo de izquierda a derecha están siendo copiados en R% de arriba a abajo (el primer dígito binario de cada número de A% va al primer elemento de R%, el segundo dígito binario de cada número de A% al segundo elemento de R%, y así sucesivamente). De esta forma, la columna vertical izquierda de dígitos binarios del carácter original, se convierte en la fila superior (horizontal) en R%, girando así el carácter en 90 grados.

Líneas 20100-20120: El carácter readaptado es transferido desde R% a A%, la matriz principal de trabajo.

Comprobación

Lance el programa, llame al menú de edición y entonces pulse «R». Tras una pausa, el carácter será vuelto a presentar, pero girado en 90 grados. Si pulsa «R» tres veces más, el carácter será vuelto a poner en su posición primitiva. Haga experimentos con combinaciones de imágenes especulares, rotaciones y caracteres inversos, hasta que se familiarice con sus efectos.

Módulo 3.1.12: Introducción en memoria de un carácter editado

Hasta ahora ha podido manipular el carácter ampliado (dentro de la celda de 8*8) hasta que quizás ha dejado de tener relación alguna con la configuración original. Sin embargo, todos estos cambios no han afectado en nada a la versión tamaño normal que se encuentra a la derecha de la celda de 8*8. Los cambios que ha hecho no se han metido todavía en la memoria de caracteres, y no se meterán hasta que no esté satisfecho con lo que ha creado. No obstante, una vez *haya* conseguido lo que desea, este módulo convertirá la configuración de la celda en parte de su conjunto de caracteres.

Módulo 3.1.12: Líneas 21000 - 21070

```
21000 REM*******************************
21010 REM Poner en memoria
21020 REM*******************************
21030 FOR i=0 TO 7
21040 c%(ca,i)=a%(i)
21050 NEXT i
21060 SYMBOL ca,a%(0),a%(1),a%(2),a%(3),
a%(4),a%(5),a%(6),a%(7)
21070 ret=1:RETURN
```

Comentarios

Línea 21040: El contenido de A% (la matriz en la que se realizan todas las manipulaciones sobre los datos del carácter) es copiado en la posición de C% que corresponde al carácter del que nos estamos ocupando.

Línea 21060: Los datos de A% se utilizan, con la potente orden SYMBOL, para asignar nuevos valores a los ocho octetos que registran la forma del carácter actual.

Comprobación

Lance el programa y mueva el puntero de carácter al carácter 65, que es la «A». Pase al modo de edición y rote el carácter una sola vez. Ahora pulse «P» y observe la pantalla. La «A» que hay a la derecha de la celda de 8*8 se transforma hasta quedar igualmente rotada, ya que el 464 está tomando la información de los caracteres del conjunto que ha creado. Es bueno recordar, cuando se editan caracteres, que a menos que desee poner las letras a su gusto resulta prudente editar tan sólo los caracteres gráficos. El hacer demasiados cambios a las letras y a los números le puede llevar a una situación en la que no pueda entender lo que el programa le está diciendo.

Módulo 3.1.13: Almacenamiento del conjunto de caracteres

Habiendo editado el conjunto de caracteres, ahora desearíamos poder guardarlo para poder utilizarlo en el futuro, pues de otra forma todo esto tendría poco sentido. Este módulo guarda en cinta su conjunto de caracteres.

Módulo 3.1.13: Líneas 22000 - 22110

```
22000 REM******************************
22010 REM Grabar en cinta
22020 REM******************************
22030 LOCATE 1,24:OPENOUT "datos caracte
res"
22040 PRINT #9,pc
22050 FOR i=pc TO 255
22060 FOR j=0 TO 7
22070 PRINT #9,c%(i,j)
22080 NEXT j
22090 NEXT i
22100 CLOSEOUT
22110 ret=1:RETURN
```

Comentarios

Líneas 22040-22090: Para ahorrar tiempo en los procesos de carga y grabado, sólo se graban los caracteres redefinidos, es decir, desde PC en adelante. No tiene sentido esperar a que se graben los datos de los 256 caracteres, si solamente se han cambiado unos 20. Los datos se graban tomando la forma de los números almacenados en C%.

Comprobación

Edite unos cuantos caracteres y métalos en memoria, después llame a este módulo para grabarlos en la cinta. La única comprobación que puede hacer por ahora, es ver que el módulo se ejecuta sin producir ningún tipo de error. Después de que se haya introducido el módulo que viene a continuación, podrá recargar el conjunto de caracteres y comprobar que se hayan grabado correctamente.

Módulo 3.1.14: Recarga del conjunto de caracteres

Una vez almacenado el conjunto de caracteres en cinta, este módulo realiza la tarea de recargarlos en memoria. Es importante que entienda este módulo ya que puede utilizarse, tal y como está, para recargar un conjunto de caracteres para otros programas. Habiendo rediseñado su conjunto de caracteres y habiéndolo almacenado en cinta, todo lo que necesita es incluir este módulo (renumerado adecuadamente si es necesario) en el programa que va a utilizar los nuevos caracteres. Reintroduzca los datos de los caracteres y rápidamente el conjunto de caracteres rediseñado quedará instalado.

Módulo 3.1.14: Líneas 23000 - 23170

```
23000 REM*******************************
23010 REM Cargar desde la cinta
23020 REM*******************************
23030 LOCATE 1,24
23040 OPENIN "datos caracteres"
23050 INPUT #9,pc
23060 FOR i=pc TO 255
23070 FOR j=0 TO 7
23080 INPUT #9,c%(i,j)
23090 NEXT j
23100 NEXT i
23110 CLOSEIN
23120 SYMBOL AFTER pc
23130 FOR i=pc TO 255
23140 SYMBOL i,c%(i,0),c%(i,1),c%(i,2),c
%(i,3),c%(i,4),c%(i,5),c%(i,6),c%(i,7)
23150 NEXT i
23160 hecho=0:ret=1
23170 RETURN
```

Comentarios

Líneas 23040-23110: Se lee el carácter de comienzo desde la cinta y, puesto que puede no corresponder al valor actual, entonces los

datos de la cinta son leídos y transferidos a C% desde la posición PC en adelante.
Líneas 23120-23150: Con los datos en C%, se utiliza SYMBOL AFTER para redefinir el conjunto de caracteres. Normalmente, esto se podría hacer durante el bucle de carga de los datos, siendo redefinido cada carácter tras haberse cogido sus ocho octetos. En la práctica, debido a lo que parece ser un error en la ROM del 464, la orden SYMBOL AFTER no es reconocida aparentemente en tanto está abierto un fichero para el Datacorder. Merecería la pena ver si este error está también presente en su máquina.

Comprobación

Pulsando «C» en el modo de edición de caracteres debería ahora poder recargar el conjunto de caracteres que grabó (mediante SAVE), como parte de la comprobación del módulo anterior. Antes de recargar lo que grabó, asegúrese de que está otra vez con el conjunto normal (el no editado) de caracteres, o sería imposible decir si se han cargado caracteres diferentes desde la cinta.

Módulo 3.1.15: Normalización del conjunto de caracteres

Es muy posible que en algún momento decida que ha ido demasiado lejos en la redefinición de sus caracteres, y que desee restaurar el conjunto de caracteres a su estado original. Esto es algo que puede hacer pulsando «X» en el menú principal; la ejecución se envía entonces al módulo de inicialización donde se le preguntará por el primer carácter a redefinir: si lo que realmente desea es volver a poner al 464 en su condición implícita, el primer carácter debería ser el 240.

Módulo 3.1.15: Líneas 24000 - 24040

```
24000 REM******************************
24010 REM Normalizar juego caracteres
24020 REM******************************
24030 GOSUB 11000
24040 hecho=0:ret=1:RETURN
```

Comprobación

Lance el programa, altere un carácter y almacene su definición en memoria. Pulse ahora «X» y observará que el carácter es restaurado a su estado original.

Módulo 3.1.16: Finalización del programa

Pulsando «T» desde el menú principal se termina el programa sin restaurar el conjunto original de caracteres.

Módulo 3.1.16: Líneas 25000 - 25030

```
25000 REM*******************************
25010 REM Terminar
25020 REM*******************************
25030 ret=1:RETURN
```

Comprobación

Altere un carácter, ponga la redefinición en memoria y termine el programa pulsando «T». Cualquier alteración que haya hecho debería seguir formando parte del conjunto de caracteres.

PROGRAMA 3.2: Diseñador

Función del programa

Sin duda, todos hemos visto las imágenes impresionantes creadas por lo que se conoce como CAD (Computer-Aided Design: diseño asistido por ordenador). Con toques hábiles el ingeniero añade líneas y formas a complejos diseños, o borra otras que ya existen. En cierta manera, aunque con limitaciones, el programa Diseñador tiene el propósito de imitar ese tipo de posibilidades. Aunque evidentemente no es tan sofisticado, le permitirá crear diseños complejos, que serán bastante mayores que el tamaño de la pantalla; para ello utiliza la pantalla de televisión, como una ventana móvil que examina zonas particulares o encoge toda el área para que pueda verse por completo. Se pueden añadir o borrar a voluntad líneas, circunferencias o rectángulos, y todo el diseño se puede almacenar en cinta para usos posteriores.

Entre los nuevos conceptos introducidos en este programa se incluyen:

1) Un cursor parpadeante (en alta resolución) definido por el usuario.
2) Almacenamiento de diseños en cinta.
3) Definición y trazado de figuras geométricas.

Módulo 3.2.1: Inicialización

Un módulo estándar de inicialización, que prepara los colores de la pantalla, dos ventanas, y una gama de variables que se explicarán a lo largo de los comentarios del programa.

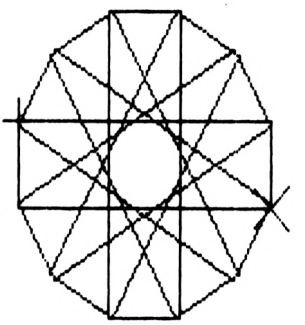


Fig. 3.3 Vaciado de pantalla del programa Diseñador.

Módulo 3.2.1: Líneas 11000 - 11250

```
11000 REM*******************************
11010 REM Inicializacion
11020 REM*******************************
11030 MODE 1
11040 BORDER 26
11050 INK 0,0
11060 INK 1,26
11070 INK 2,18
11080 INK 3,18
11090 PAPER 1:PEN 0
11100 CLS
11110 WINDOW 27,40,1,25
11120 ORIGIN 200,200,0,398,398,0
11130 CLG
11140 WINDOW #1,27,40,25,25:PAPER #1,1:P
EN #1,0
11150 cambio$=CHR$(23)+CHR$(1)
11160 normal$=CHR$(23)+CHR$(0)
11170 menu$=""
11180 FOR i=240 TO 247:menu$=menu$+CHR$(
i):NEXT i
11190 menu$=menu$+"12pclr"+CHR$(13)+"svg
b\"
11200 unidad=1
11210 pix=2
```

```
11220 izq=-100:der=99
11230 sup=99:inf=-100
11240 DIM a%(1000,4),x(1),y(1)
11250 RETURN
```

Módulo 3.2.2: Dos cursores de trazado

En cualquier programa de diseño, una de las primeras necesidades es que el usuario debe saber dónde está la posición de trazado en cada momento. Esto se consigue normalmente por medio de un cursor de algún tipo, que marque la posición actual sobre la pantalla. Este programa utiliza dos cursores para definir, por ejemplo, los dos extremos de una línea que se desea trazar, las dos esquinas opuestas de un rectángulo, etc. Ambos cursores pueden ser movidos sobre el diseño, mediante las teclas de control de cursor, sin afectar al contenido de la pantalla.

Módulo 3.2.2: Líneas 12000 - 13080

```
12000 REM*******************************
12010 REM Cursor 1
12020 REM*******************************
12030 PRINT cambio$
12040 PLOT x(0)*pix-16,y(0)*pix,2
12050 DRAWR 32,0
12060 PLOT x(0)*pix,y(0)*pix-16
12070 DRAWR 0,32
12080 RETURN
13000 REM*******************************
13010 REM Cursor 2
13020 REM*******************************
13030 PRINT cambio$
13040 PLOT x(1)*pix-16,y(1)*pix-16,2
13050 DRAWR 32,32
13060 PLOT x(1)*pix-16,y(1)*pix+16
13070 DRAWR 32,-32
13080 RETURN
```

Comentarios

Líneas 12030 y 13030: En el módulo de inicialización se definió CAMBIO$ como CHR$(23) más CHR$(1). CHR$(23) es un carácter de control, un carácter que no presenta nada en pantalla pero que tiene un efecto sobre la forma en que trabaja el 464.

En este caso particular, el efecto del carácter es activar lo que se

conoce como una característica de «CAMBIO» en el trazado por la pantalla o, más técnicamente, activar un XOR (modo OR eXclusivo). Lo que esto significa es que cualquier cosa que se trace en pantalla, mientras está activada la característica de CAMBIO, cambiará al estado opuesto a cualquier pixel con el que se tropiece; si tenían el color de primer plano se les cambiará al color de fondo y viceversa. La ventaja de esto es que, si se traza *dos* veces la misma figura con el CAMBIO activado, la pantalla quedará exactamente en las mismas condiciones con las que se comenzó. En otras palabras, se puede trazar un cursor y entonces retrazarlo para borrarlo, todo ello sin tener el menor efecto, a largo plazo, sobre cualquier cosa que esté en la pantalla.
Líneas 12040-12070 y 13040-13070: Se trazan los cursores, donde uno de ellos es una cruz formada por una raya horizontal y otra vertical, y el segundo es otra cruz en la que dichas rayas forman 45 grados respecto a las del primero. En cada caso, la posición de la cruz está determinada por las matrices (de dos elementos) X e Y, junto con el valor de la variable PIX. La razón por la que se necesita PIX es que cuando avance por el programa encontrará que el diseño se puede mirar a escalas diferentes, de forma que una cierta distancia en un diseño puede variar cuando se presenta en pantalla; la variable PIX asegura que el cursor está en la posición correcta sea cual sea la escala a la que se mire el diseño.

Módulo 3.2.3: Selección del cursor

Con dos cursores va a ser obviamente necesario poder seleccionar el que debe ser direccionado. Este módulo es llamado desde el menú principal, que introduciremos en breve, para hacer justamente eso, cosa que se consigue cambiando el valor de la variable NC (Número de Cursor).

Módulo 3.2.3: Líneas 23000 - 24040

```
23000 REM*******************************
23010 REM Seleccionar cursor 1
23020 REM*******************************
23030 nc=0
23040 RETURN
24000 REM*******************************
24010 REM Seleccionar cursor 2
24020 REM*******************************
24030 nc=1
24040 RETURN
```

Módulo 3.2.4: Movimientos de los cursores

Por último, necesitamos poder mover los cursores. Esto se hace por medio de las teclas de cursor, que mueven el cursor en una unidad de pantalla y en la dirección adecuada (cuando no se tiene pulsada la tecla de cambio, la de minúsculas a mayúsculas), en tanto que lo mueven en 16 unidades si dicha tecla está pulsada. Cada una de las rutinas de este módulo es llamada, por separado, desde el menú, cuando se pulsa la tecla apropiada. Esto hace que el programa sea más largo, pero acelera el movimiento de los cursores, cuando se le compara con una sub-subrutina llena de sentencias IF para ocuparse de las teclas de cursor.

Módulo 3.2.4: Líneas 15000 - 22040

```
15000 REM*******************************
15010 REM Cursor arriba 1 unidad
15020 REM*******************************
15030 y(nc)=y(nc)+unidad
15040 RETURN
16000 REM*******************************
16010 REM Cursor abajo 1 unidad
16020 REM*******************************
16030 y(nc)=y(nc)-unidad
16040 RETURN
17000 REM*******************************
17010 REM Cursor a la izq. 1 unidad
17020 REM*******************************
17030 x(nc)=x(nc)-unidad
17040 RETURN
18000 REM*******************************
18010 REM Cursor a la der. 1 unidad
18020 REM*******************************
18030 x(nc)=x(nc)+unidad
18040 RETURN
19000 REM*******************************
19010 REM Cursor arriba 16 unidades
19020 REM*******************************
19030 y(nc)=y(nc)+unidad*16
19040 RETURN
20000 REM*******************************
20010 REM Cursor abajo 16 unidades
20020 REM*******************************
20030 y(nc)=y(nc)-unidad*16
20040 RETURN
21000 REM*******************************
21010 REM Cursor a la izq. 16 unidades
```

```
21020 REM*******************************
21030 x(nc)=x(nc)-unidad*16
21040 RETURN
22000 REM*******************************
22010 REM Cursor a la der. 16 unidades
22020 REM*******************************
22030 x(nc)=x(nc)+unidad*16
22040 RETURN
```

Módulo 3.2.5: Introducción de órdenes

Habiéndonos proporcionado dos cursores y la capacidad de moverlos, comenzamos con el módulo que asigna los trabajos a las diversas partes del programa cuando se pulsa una tecla. Su funcionamiento correcto, sólo podrá ser comprobado cuando introduzcamos el módulo siguiente (el breve módulo de control). Por supuesto, que la mayoría de las órdenes individuales, mencionadas en los comentarios, no serán operativas hasta que se añadan módulos posteriores.

Módulo 3.2.5: Líneas 14000 - 14440

```
14000 REM*******************************
14010 REM Modo edicion
14020 REM*******************************
14030 CLS
14040 PRINT " MODO EDICION":PRINT
14050 PRINT "X1:"
14060 PRINT "Y1:":PRINT
14070 PRINT "X2:"
14080 PRINT "Y2:":PRINT
14090 PRINT "Escala:";unidad
14100 PRINT "Figuras:";figura:PRINT
14110 PRINT "'1' Cursor 1"
14120 PRINT "'2' Cursor 2"
14130 PRINT "'P' Poligono"
14140 PRINT "'C' Circunf."
14150 PRINT "'L' Linea"
14160 PRINT "'R' Rectang."
14170 PRINT "'";CHR$(1);CHR$(13);"' Fija
r"
14180 PRINT "'S' Suprimir"
14190 PRINT "'V' Ventana"
14200 PRINT "'G' Grabar"
14210 PRINT "'B' Borrado"
14220 PRINT "'\' Terminar"
```

```
14230 t=0:WHILE t<19
14240 GOSUB 12000
14250 GOSUB 13000
14260 LOCATE 4,3:PRINT x(0);TAB(14)
14270 LOCATE 4,4:PRINT y(0);TAB(14)
14280 LOCATE 4,6:PRINT x(1);TAB(14)
14290 LOCATE 4,7:PRINT y(1);TAB(14)
14300 t=0:WHILE t=0
14310 t$="":WHILE t$=""
14320 t$=LOWER$(INKEY$)
14330 WEND
14340 t=INSTR(menu$,t$)
14350 WEND
14360 GOSUB 12000
14370 GOSUB 13000
14380 ON t GOSUB 15000,16000,17000,18000
,19000,20000,21000,22000,23000,24000,250
00,26000,27000,28000,29000,30000,31000,3
2000
14390 IF x(nc)>der THEN x(nc)=der
14400 IF x(nc)<izq THEN x(nc)=izq
14410 IF y(nc)>sup THEN y(nc)=sup
14420 IF y(nc)<inf THEN y(nc)=inf
14430 WEND
14440 RETURN
```

Comentarios

Líneas 14030-14290: El menú del programa que a causa de la forma en que se preparó la pantalla en el módulo de inicialización, se presenta en la parte derecha de la pantalla (la parte central y la izquierda están reservadas para el diseño en sí). Además del menú se trazan los cursores sobre la zona de diseño, y se listan sus posiciones.

Líneas 14300-14350: Estos bucles anidados esperan a que se pulse una tecla, y entonces la comparan con la cadena MENU$ que se preparó en el módulo de inicialización, generando un valor para la orden ON...GOSUB de la línea 14380.

Líneas 14390-14420: Estas líneas corrigen las coordenadas del cursor en caso de que, como resultado de una orden de movimiento de cursor, dicho cursor se mueva fuera de los límites de la ventana en vigor (registrada en las variables INF, SUP, IZQ y DER).

Módulo 3.2.6: El módulo de control

En este punto es necesario introducir el breve módulo de control para que se puedan comprobar las funciones de movimiento de cur-

sor, así como los módulos siguientes conforme se vayan introduciendo. Observe que el módulo proporciona la posibilidad de cargar datos desde la cinta, cosa que se explicará en detalle cuando sean introducidos los módulos pertinentes.

Módulo 3.2.6: Líneas 10000 - 10100

```
10000 REM******************************
10010 REM Control
10020 REM******************************
10030 INPUT "Desea cargar desde la cinta
 (s/n)";q$
10040 GOSUB 11000
10050 IF LOWER$(q$)="s" THEN GOSUB 33000
10060 WHILE 1
10070 GOSUB 14000
10080 IF t=20 THEN MODE 1:CLS:END
10090 GOSUB 40000
10100 WEND
```

Comprobación

Ejecute el programa, y observará que puede mover cualquiera de los cursores por medio de las teclas de control de cursor. Ninguna de las otras funciones del programa estarán aún disponibles.

Módulo 3.2.7: Trazado de una línea

Ahora viene una serie de tres módulos que hacen el trabajo pesado del programa, trazando respectivamente: líneas, rectángulos y polígonos (incluyendo circunferencias). No podrá utilizarlos inmediatamente, ya que estos módulos no son llamados directamente desde el menú, sino por otros módulos que tienen la tarea de gestionar el trabajo de trazado/borrado y demás. Este módulo traza simplemente una línea desde la posición del cursor 2 hasta la posición del cursor 1.

Módulo 3.2.7: Líneas 37000 - 37050

```
37000 REM******************************
37010 REM Trazar linea
37020 REM******************************
37030 PLOT x2*pix,y2*pix,c
37040 DRAW x1*pix,y1*pix
37050 RETURN
```

Comprobación

Cambie provisionalmente la línea 10080 por:

```
10080 IF t=20 THEN CLS:END
```

Lance el programa para preparar la pantalla, y pulse «\» para terminar, después teclee:

```
C=2:X1=50:Y1=50:GOTO 37000[ENTER]
```

con lo que debería ver una línea trazada desde el centro del área negra, en sentido diagonal, hacia arriba y hacia la derecha.

Vuelva a poner la línea 10080 en su forma original.

Módulo 3.2.8: Trazado de una circunferencia o de un polígono

Se trata de un módulo de uso general, que traza líneas entre puntos que pertenecen a una circunferencia. Dependiendo del número de lados especificado, el contorno trazado aparecerá como una circunferencia o como un polígono regular (si se utilizan tres puntos, la figura será un triángulo).

Módulo 3.2.8: Líneas 38000 - 38140

```
38000 REM*******************************
38010 REM Trazar poligono
38020 REM*******************************
38030 r=SQR((x1-x2)*(x1-x2)+(y1-y2)*(y1-
y2))
38040 z2=CINT(8*LOG(r/unidad))
38050 IF NOT(z1=2 OR z1>z2) THEN z2=z1
38060 IF x1=x2 THEN a=PI/2*SGN(y1-y2)
38070 IF x1<>x2 THEN a=ATN((y1-y2)/(x1-x
2))
38080 IF x1<x2 THEN a=a+PI
38090 PLOT (x2+r*COS(a))*pix,(y2+r*SIN(a
))*pix,c
38100 FOR s=1 TO z2
38110 a1=a+2*PI*s/z2
38120 DRAW (x2+r*COS(a1))*pix,(y2+r*SIN(
a1))*pix
38130 NEXT s
38140 RETURN
```

Comentarios
Línea 38030: La distancia entre los dos cursores.
Líneas 38040-38050: Z2 será el número de puntos a trazar en torno a la circunferencia. La expresión de la línea 38040 proporciona suficientes puntos como para que la circunferencia dibujada sea aceptable (no tiene mucho sentido definir demasiados puntos, ya que no habrá una mejora apreciable, y sin embargo el proceso puede hacerse terriblemente lento). Cuando el usuario especifica una figura distinta a una circunferencia, la línea 38050 fija el número de puntos apropiado a lo que se ha especificado (supuesto que este número no sea mayor que el necesario para trazar la circunferencia).
Líneas 38060-38080: Estas tres líneas calculan el ángulo del cursor 1 en relación con el cursor 2.
Líneas 38090: El primer punto en torno a la circunferencia, que debería ser casi idéntico a la posición del cursor 2. Debería reconocer la ecuación del programa Analoj del capítulo 1.
Líneas 38100-38130: Este bucle calcula la posición de Z2 puntos en torno a la periferia del círculo, trazando una línea a cada uno según se van calculando. Una vez más, la fórmula es la misma que la explicada en el programa Analoj.

Comprobación
Para comprobar este módulo y el siguiente hay que ajustar tantas variables, que es mejor esperar hasta que se hayan introducido más módulos para parchearlos dentro de la secuencia principal del programa.

Módulo 3.2.9: Trazado de un rectángulo

La última figura que es capaz de trazar el programa es un simple rectángulo, con sus esquinas opuestas situadas en las posiciones de los cursores. Este módulo es ligeramente más complejo que los anteriores, puesto que no hay orden incorporada para la figura y, además, hemos incluido la posibilidad de rotarla.

Módulo 3.2.9: Líneas 36000 - 36210

```
36000 REM*******************************
36010 REM Trazar rectangulo
36020 REM*******************************
36030 a=-z1/10000
36040 cs=COS(a)
36050 sn=SIN(a)
36060 x=(x1+x2)/2
```

```
36070 y=(y1+y2)/2
36080 rx1=x+(x1-x)*cs+(y1-y)*sn
36090 ry1=y+(y1-y)*cs-(x1-x)*sn
36100 rx2=x+(x2-x)*cs+(y1-y)*sn
36110 ry2=y+(y1-y)*cs-(x2-x)*sn
36120 rx3=x+(x2-x)*cs+(y2-y)*sn
36130 ry3=y+(y2-y)*cs-(x2-x)*sn
36140 rx4=x+(x1-x)*cs+(y2-y)*sn
36150 ry4=y+(y2-y)*cs-(x1-x)*sn
36160 PLOT rx1*pix,ry1*pix,c
36170 DRAW rx2*pix,ry2*pix
36180 DRAW rx3*pix,ry3*pix
36190 DRAW rx4*pix,ry4*pix
36200 DRAW rx1*pix,ry1*pix
36210 RETURN
```

Comentarios
Líneas 36030-36050: La variable A contiene el ángulo, suministrado por el usuario, a que debe ser girado el rectángulo. CS y SN se utilizarán para abreviar las fórmulas del movimiento de las esquinas cuando se gira el rectángulo.
Líneas 36080-36150: Las fórmulas para rotar un punto en torno a otro son las siguientes:

X2=XO+XD*COS(ANGULO)+YD*SIN(ANGULO)
Y2=YO+YD*COS(ANGULO)–XD*SIN(ANGULO)

donde X2 e Y2 son las coordenadas que resultan al rotar el punto X, Y; XO e YO son las coordenadas en torno a las que se ha girado el punto; XD e YD son las distancias desde X e Y hasta XO e YO respectivamente, y ANGULO es el ángulo bajo el que se gira el punto definido por X e Y.

El rectángulo básico, sin rotar, tendrá sus esquinas en X1/Y1, X2/Y1, X2/Y2 y X1/Y2. Mirando las líneas y comparándolas con las fórmulas anteriores, debería ver que proporcionan las coordenadas de las cuatro esquinas una vez rotadas.

Módulo 3.2.10: Asignación del trabajo entre los módulos de trazado

Este breve módulo es uno de los necesarios para parchear los módulos de trazado con lo que viene a continuación. Su uso se explicará cuando examinemos el módulo siguiente.

Módulo 3.2.10: Líneas 35000 - 35060

```
35000 REM*******************************
35010 REM Trazado global
35020 REM*******************************
35030 IF z1<1 THEN GOSUB 36000
35040 IF z1=1 THEN GOSUB 37000
35050 IF z1>1 THEN GOSUB 38000
35060 RETURN
```

Módulo 3.2.11: Control de las órdenes de trazado

Llegamos ahora a las tres subrutinas que traducen las pulsaciones individuales, desde el módulo de menú, en los parámetros necesarios para usar los módulos de trazado concretos.

Cada uno de los módulos tiene las funciones siguientes:

1) En caso necesario solicitan más información, como: «¿Cuántos lados?» o «¿Grados a rotar?».
2) Activan la característica de CAMBIO imprimiendo CAMBIO$.
3) Borran cualquier figura existente que no haya sido confirmada por el usuario, según es indicado por la variable TEMP al ser igual a 1.
4) Preparan las variables Z1, X1, Y1, X2 e Y2, las posiciones de los cursores.
5) Llaman al módulo de trazado pertinente.
6) Activan la variable TEMP, para mostrar que se ha trazado una figura y aún no se ha confirmado.

Módulo 3.2.11: Líneas 25000 - 28110

```
25000 REM*******************************
25010 REM Poligono
25020 REM*******************************
25030 z=0:e=0:WHILE z<3 OR z>32767
25040 IF e=1 THEN x$="**NO  VALIDO**":GO
SUB 34000
25050 INPUT #1,"Lados";z:PRINT #1
25060 e=1:WEND
25070 PRINT cambio$:c=2
25080 IF temp=1 THEN GOSUB 35000
25090 z1=z:x1=x(0):y1=y(0):x2=x(1):y2=y(
1)
25100 GOSUB 38000
```

```
25110 temp=1
25120 RETURN
26000 REM*******************************
26010 REM Circunferencia
26020 REM*******************************
26030 PRINT cambio$:c=2
26040 IF temp=1 THEN GOSUB 35000
26050 z1=2:x1=x(0):y1=y(0):x2=x(1):y2=y(
1)
26060 GOSUB 38000
26070 temp=1
26080 RETURN
27000 REM*******************************
27010 REM Linea
27020 REM*******************************
27030 PRINT cambio$:c=2
27040 IF temp=1 THEN GOSUB 35000
27050 z1=1:x1=x(0):y1=y(0):x2=x(1):y2=y(
1)
27060 GOSUB 37000
27070 temp=1
27080 RETURN
28000 REM*******************************
28010 REM Rectangulo
28020 REM*******************************
28030 INPUT #1,"Rotacion";a:PRINT #1
28040 a=a-180*INT(a/180)
28050 z=-CINT(a/180*PI*10000)
28060 PRINT cambio$:c=2
28070 IF temp=1 THEN GOSUB 35000
28080 z1=z:x1=x(0):y1=y(0):x2=x(1):y2=y(
1)
28090 GOSUB 36000
28100 temp=1
28110 RETURN
```

Comentarios

En vez de comentar cada subrutina, se analizarán brevemente los puntos principales de una de ellas, puntos que se aplican a las restantes.

Líneas 25030-25060: El resto de la información necesaria para trazar una cierta figura, por ejemplo: ¿Lados?

Línea 25080: Si la variable TEMP está activada (puesta a 1), entonces se acaba de trazar una figura y no ha sido confirmada (en breve consideraremos las confirmaciones). Antes de trazar la figura en vigor se utiliza el módulo de la línea 35000 para llamar otra vez al módulo de

trazado pertinente. Puesto que la característica de CAMBIO está activada, el módulo dibujará sobre la figura anterior y la borrará. Lo que esto significa es que, si el usuario desea trazar una figura en un diseño complejo, y no le sale totalmente bien (no cuadra perfectamente), puede mover uno de los cursores, usando esta figura poco satisfactoria como guía, y trazar entonces otra vez la figura borrando automáticamente la primera versión.
Línea 25090: Se preparan las variables.

Comprobación

Ahora debería poder lanzar el programa y trazar figuras, aunque aún no puede confirmarlas y convertirlas en parte permanente del diseño (cada figura trazada borrará la anterior).

Módulo 3.2.12: Registro de un diseño

Un programa como este tiene poca utilidad, excepto la de pasar un rato divertido, a menos que el diseño en el que se está trabajando se pueda registrar de alguna manera. En nuestro caso, el registro del diseño en una matriz nos permitirá, posteriormente, tanto almacenar los datos en cinta como suprimir líneas individuales. Las líneas y las figuras van siendo registradas al llamar a este módulo conforme se las introduce. Además, el módulo traza las figuras de forma permanente, con lo que no serán borradas al dibujar las siguientes. El módulo es llamado al pulsar ENTER en el menú principal.

Módulo 3.2.12: Líneas 29000 - 29150

```
29000 REM*******************************
29010 REM Fijar
29020 REM*******************************
29030 IF temp=0 THEN RETURN
29040 IF figura=1001 THEN x$="NO MAS ESP
ACIO":GOSUB 34000:RETURN
29050 PRINT normal$:c=1
29060 GOSUB 35000
29070 a%(figura,0)=z1
29080 a%(figura,1)=x1
29090 a%(figura,2)=y1
29100 a%(figura,3)=x2
29110 a%(figura,4)=y2
29120 temp=0
29130 figura=figura+1
29140 LOCATE 9,10:PRINT figura;TAB(14)
29150 RETURN
```

Comentarios
Línea 29030: Si TEMP no es igual a 1, no hay ninguna figura a registrar.
Líneas 29050-29060: La cadena NORMAL$, que se definió en el módulo de inicialización, desactiva la característica de CAMBIO, de manera que se trace lo que se trace se hará en la forma normal. Se llama entonces al módulo de la línea 35000 para elegir entre los diversos módulos de trazado.
Líneas 29070-29130: Se almacenan en la matriz A% las variables necesarias para trazar la figura. TEMP se pone a 0 para indicar que no hay ninguna figura sin confirmar sobre la pantalla, y se incrementa la variable FIGURA para mostrar que se ha añadido otra figura al diseño.

Módulo 3.2.13: Supresión de una figura

Un módulo sencillo para borrar una figura no confirmada de la misma forma que cuando se traza una nueva figura.

Módulo 3.2.13: Líneas 30000 - 30070

```
30000 REM*******************************
30010 REM Suprimir
30020 REM*******************************
30030 IF temp=0 THEN RETURN
30040 PRINT cambio$;c=2
30050 GOSUB 35000
30060 temp=0
30070 RETURN
```

Comprobación
Ahora debería poder suprimir (borrar) una figura no confirmada pulsando «S» en el menú.

Módulo 3.2.14: Retrazado de un diseño

Teniendo la capacidad de registrar una figura en una matriz, pasamos ahora a la cuestión del retrazado de todo el diseño, partiendo de los datos almacenados en la matriz y volviendo a colocarlos en la pantalla.

Módulo 3.2.14: Líneas 39000 - 39130

```
39000 REM*******************************
39010 REM Volver a trazar
39020 REM*******************************
39030 IF figura=0 THEN RETURN
39040 PRINT normal$:c=1
39050 FOR i=0 TO figura-1
39060 z1=a%(i,0)
39070 x1=a%(i,1)
39080 y1=a%(i,2)
39090 x2=a%(i,3)
39100 y2=a%(i,4)
39110 GOSUB 35000
39120 NEXT i
39130 RETURN
```

Comprobación

Cambie la línea 10080 por:

```
10080 IF t=20 THEN CLS:END
```

Lance el programa y trace unas cuantas figuras, detenga entonces el programa mediante «\», y teclee:

```
CLS #2[ENTER]
GOTO 39000[ENTER]
```

con lo que debería ver todas las figuras vueltas a trazar en la pantalla.

No olvide volver a poner la línea 10080 en su estado original.

Módulo 3.2.15: Borrado de líneas y figuras

Puesto que el diseño no se almacena como un todo, sino en forma de líneas y figuras individuales, resulta sencillo dar al usuario la opción de borrar dichas figuras. Este módulo presenta todo el diseño, línea por línea, con la opción de borrar cualquier línea. Se puede utilizar también para redibujar todo el diseño, en caso de que se haya borrado la pantalla por detener el programa.

Módulo 3.2.15: Líneas 40000 - 40290

```
40000 REM*******************************
40010 REM Modo Borrado
40020 REM*******************************
```

```
40030 CLS
40040 CLG
40050 PRINT " MODO BORRADO":PRINT
40055 i=1
40060 PRINT "Figura #";i:PRINT
40070 PRINT "Escala:";unidad
40080 PRINT "Figuras:";figura:PRINT:PRIN
T
40090 PRINT "'";CHR$(1);CHR$(13);"' Figu
ra":PRINT "    siguiente":PRINT
40100 PRINT "'S' Suprimir":PRINT
40110 PRINT "'\' Volver a":PRINT "    Mo
do":PRINT "    Edicion"
40120 t$="":i=0:WHILE i<figura
40130 z1=a%(i,0)
40140 x1=a%(i,1)
40150 y1=a%(i,2)
40160 x2=a%(i,3)
40170 y2=a%(i,4)
40180 d=0:WHILE t$<>"\" AND d=0
40190 PRINT cambio$;c=2
40200 GOSUB 35000
40210 t$=""
40220 d=1:WEND
40230 WHILE t$=""
40240 t$=UPPER$(INKEY$)
40250 WEND
40260 IF t$="S" THEN GOSUB 41000 ELSE GO
SUB 42000
40270 WEND
40280 temp=0
40290 RETURN
```

Comentarios

Líneas 40120-40220: Desde la matriz A% se recuperan los detalles de la figura, figura que se dibuja con la característica de CAMBIO activada.

Líneas 40230-40280: El programa espera a que el usuario pulse «S» para borrar la figura presentada, o cualquier otra tecla para confirmarla. Ninguna de las dos opciones tendrán aún efecto, ya que es necesario introducir antes los dos módulos siguientes.

Módulo 3.2.16: Supresión de una figura en la matriz

Una vez retrazada una figura a partir de la matriz, su borrado permanente del diseño sólo requiere que se trace otra vez con la ca-

racterística de CAMBIO activada, y colapsar entonces la matriz para eliminar el registro de sus detalles.

Módulo 3.2.16: Líneas 41000 - 41120

```
41000 REM*******************************
41010 REM Suprimir
41020 REM*******************************
41030 PRINT cambio$:c=2
41040 GOSUB 35000
41050 figura=figura-1
41060 FOR j=i TO figura-1
41070 FOR k=0 TO 4
41080 a%(j,k)=a%(j+1,k)
41090 NEXT k
41100 NEXT j
41110 LOCATE 9,6:PRINT figura;TAB(14)
41120 RETURN
```

Módulo 3.2.17: Confirmación de una figura en el modo de borrado

Para confirmar una figura, todo lo que se necesita es volver a dibujarla con la característica de CAMBIO desactivada.

Módulo 3.2.17: Líneas 42000 - 42070

```
42000 REM*******************************
42010 REM Figura siguiente
42020 REM*******************************
42030 PRINT normal$:c=1
42040 GOSUB 35000
42050 i=i+1
42060 LOCATE 9,3:PRINT i+1
42070 RETURN
```

Comprobación

Supuesto que haya introducido algunas figuras, debería poder pulsar «B», en la parte principal del programa, para acceder a este módulo. Compruebe que puede recorrer las figuras del diseño que ha introducido, borrándolas o dejándolas sin tocar. La pulsación de «\» al principio del diseño no debería dar lugar a ningún cambio, sino al retrazado del diseño íntegro.

Módulo 3.2.18: Mensajes de error

Un módulo sencillo para presentar un mensaje de error especificado por otra parte del programa.

Módulo 3.2.18: Líneas 34000 - 34070

```
34000 REM*******************************
34010 REM Error
34020 REM*******************************
34030 PRINT #1,x$;
34040 SOUND 1,1000,100
34050 FOR i=1 TO 1500:NEXT i
34060 PRINT #1
34070 RETURN
```

Módulo 3.2.19: Ventanas y escalas

Hasta ahora no nos hemos ocupado en absoluto de una de las propiedades más útiles de Diseñador: mover la pantalla como una ventana sobre un gran diseño o contraerlo para que pueda verse por completo en una única pantalla. Se trata de un módulo bastante complejo, pero podía haber sido aún peor si no fuera por el hecho de que el 464 hace muy fácil mover el origen (ORIGIN) de gráficos, y dibujar líneas que no aparecen en pantalla, sin generar ningún error.

Módulo 3.2.19: Líneas 31000 - 31280

```
31000 REM*******************************
31010 REM Ventana
31020 REM*******************************
31030 x$="**NO  VALIDO**"
31040 e=0:WHILE e=0 OR x<-16384 OR x>163
83
31050 IF e=1 THEN GOSUB 34000
31060 INPUT #1,"X";x
31070 e=1:WEND
31080 e=0:WHILE e=0 OR y<-16384 OR y>163
83
31090 IF e=1 THEN GOSUB 34000
31100 INPUT #1,"Y";y
31110 e=1:WEND
31120 e=0:WHILE e=0 OR unidad<1
31130 IF e=1 THEN GOSUB 34000
```

```
31140 INPUT #1,"Escala";unidad:PRINT #1
31150 e=1:WEND
31160 pix=2/unidad
31170 LOCATE 8,9:PRINT unidad;TAB(14)
31180 izq=x-100*unidad:IF izq<-16384 THE
N izq=-16384
31190 der=x+99*unidad:IF der>16383 THEN
der=16383
31200 sup=y+99*unidad:IF sup>16383 THEN
sup=16383
31210 inf=y-100*unidad:IF inf<-16384 THE
N inf=-16384
31220 x(0)=x:y(0)=y
31230 x(1)=x:y(1)=y
31240 temp=0
31250 ORIGIN 200-x*pix,200-y*pix,0,398,3
98,0
31260 CLG
31270 GOSUB 39000
31280 RETURN
```

Comentarios

Líneas 31030 y 31050: Una llamada a la rutina de mensaje de error que acaba de introducirse. El módulo tiene varias llamadas para avisar al usuario de que se han introducido cifras incorrectas.

Líneas 31060-31110: Se le pide al usuario que introduzca las coordenadas X e Y del centro de la nueva ventana. El tamaño total del diseño es de 32768*32768 unidades, y la ventana puede moverse por cualquier zona de esta área teórica.

Líneas 31120-31150: Se introduce la escala de la ventana. Cuanto más grande sea la cifra introducida, más pequeño será el diseño cuando se presente en pantalla. Esto permite crear un diseño a una escala que sea bastante más grande que la pantalla y visualizarlo, entonces, a una escala reducida, de forma que todo él pueda verse de una vez.

Líneas 31180-31210: Se fijan las variables que representan los límites de la pantalla. Si cualquiera de los límites cae fuera del tamaño máximo del diseño, la ventana es desplazada para colocarla dentro de los márgenes.

Líneas 31220-31270: Ambos cursores son trasladados al centro de la nueva ventana, y se restaura el origen de gráficos. La pantalla de gráficos se borra, y se llama entonces al módulo de retrazado para que vuelva a crear el diseño. Es muy posible que, con la ventana movida, parte del diseño, o todo él, caiga fuera de la nueva ventana y no sea, por tanto, visible.

Comprobación

Con un diseño introducido, debería poder mover la pantalla sobre él y ampliarlo o encogerlo a voluntad.

Módulo 3.2.20: Grabación y carga en y desde la cinta

Una sección estándar de almacenamiento de datos. Si desea aclaraciones sobre los métodos empleados, vea el programa Gráficos-3D del capítulo anterior.

Módulo 3.2.20: Líneas 32000 - 33130

```
32000 REM*******************************
32010 REM Grabar en cinta
32020 REM*******************************
32030 INPUT #1,"Nombre:",n$:PRINT #1
32040 OPENOUT "!"+n$
32050 PRINT #9,figura
32060 FOR i=0 TO figura-1
32070 FOR j=0 TO 4
32080 PRINT #9,a%(i,j)
32090 NEXT j
32100 NEXT i
32110 CLOSEOUT
32120 RETURN
33000 REM*******************************
33010 REM Cargar desde la cinta
33020 REM*******************************
33030 INPUT #1,"Nombre:",n$:PRINT #1
33040 OPENIN "!"+n$
33050 INPUT #9,figura
33060 FOR i=0 TO figura-1
33070 FOR j=0 TO 4
33080 INPUT #9,a%(i,j)
33090 NEXT j
33100 NEXT i
33110 CLOSEIN
33120 GOSUB 39000
33130 RETURN
```

Comprobación

Cree un diseño sencillo y utilice «G» para grabarlo. Pare el programa y láncelo de nuevo para borrar la memoria. Conteste «S» a la pregunta de si desea cargar desde la cinta. Dé el nombre del fichero bajo el que fue grabado el diseño, y debería ver dicho diseño vuelto a crear en pantalla.

Si esta prueba tiene éxito, el programa está listo para ser utilizado.

PROGRAMA 3.3: Música

Función del programa

Aparte de sus muchas otras aptitudes, el 464 es indudablemente uno de los micros musicales más potentes de la generación actual, con una gama de recursos prácticos que la mayoría de los propietarios de otros micros sólo pueden soñar. No sólo eso, contrariamente a algunas otras máquinas, de recursos igualmente potentes, el BASIC del 464 está concebido para permitir que cualquiera, con un esfuerzo mínimo, pueda sacar el máximo provecho de la potencia musical que proporciona el hardware.

Este programa es un buen ejemplo de lo que se puede conseguir con un poco de reflexión. Usando el programa, podrá preparar complejas melodías y armonías, y manipularlas de diversas formas. Tanto si tiene preparación musical como si solamente es un diletante, esperamos que los resultados que proporciona el programa le sorprenderán y deleitarán.

```
          CAMBIAR PARAMETROS
          ==================

Pulse ENTER
 para dejar parametro inalterado

Octava relativa ( 0 ) :1
Nota relativa ( 0 ) :3
Duracion unitaria ( 20 ) :15
Tocar canal 0 (S) :
Tocar canal 1 (S) :
Tocar canal 2 (S) :N
```

Fig. 3.4 Un menú del programa Música.

Módulo 3.3.1: Los datos de la melodía

Como en los programas de gráficos que se dieron anteriormente, el método adoptado aquí es colocar, al final del programa, la información en la que se basará la melodía en una sección de sentencias

data, permitiéndose así una facilidad muchísimo mayor de revisión y de edición. Por ahora no es importante comprender los datos que aparecen, ya que sus diversos aspectos se explicarán conforme avancemos por las partes del programa que hacen uso de ellos. Los datos concretos que se presentan aquí, harán que suene una interpretación bastante decente de «Fur Elise».

Módulo 3.3.1: Líneas 26000 - 28390

```
26000 REM*******************************
26010 REM Datos canal 0
26020 REM*******************************
26030 DATA ENV1
26040 DATA *
26050 DATA D1,16,15
26060 DATA 16,15,16,11,14,12
26070 DATA D2,9,D1,R,0,4,9
26080 DATA D2,11,D1,R,4,8,11
26090 DATA D2,12,D1,R,4,16,15
26100 DATA 16,15,16,11,14,12
26110 DATA D2,9,D1,R,0,4,9
26120 DATA D2,11,D1,R,4,8,11
26130 DATA [1
26140 DATA D4,9
26150 DATA ]
26160 DATA *2
26170 DATA [2
26180 DATA D2,9,D1,R,11,12,14
26190 DATA *
26200 DATA D3,16,D1,7,17,16
26210 DATA D3,14,D1,5,16,14
26220 DATA D3,12,D1,4,14,12
26230 DATA D2,11,D1,R,4,16,R
26240 DATA R,16,28,R2,15
26250 DATA 16,R2,15,16,15
26260 DATA 16,15,16,11,14,12
26270 DATA D2,9,D1,R,0,4,9
26280 DATA D2,11,D1,R,4,8,11
26290 DATA D2,12,D1,R,4,16,15
26300 DATA 16,15,16,11,14,12
26310 DATA D2,9,D1,R,0,4,9
26320 DATA D2,11,D1,R,4,12,11
26330 DATA [1
26340 DATA D2,9,D1,R,11,12,14
26350 DATA ]
26360 DATA *2
```

```
26370 DATA D2,9,D1,R,0,4,9
26380 DATA 12,16,D4,21
26390 DATA fin
27000 REM*******************************
27010 REM Datos canal 1
27020 REM*******************************
27030 DATA fin
28000 REM*******************************
28010 REM Datos canal 2
28020 REM*******************************
28030 DATA ENV1,0-2
28040 DATA *
28050 DATA R2
28060 DATA R6
28070 DATA 9,16,21,R3
28080 DATA 4,16,20,R3
28090 DATA 9,16,21,R3
28100 DATA R6
28110 DATA 9,16,21,R3
28120 DATA 4,16,20,R3
28130 DATA [1
28140 DATA 9,16,21,R1
28150 DATA ]
28160 DATA *2
28170 DATA [2
28180 DATA 9,16,21,R3
28190 DATA *
28200 DATA 12,19,24,R3
28210 DATA 7,19,23,R3
28220 DATA 9,16,21,R3
28230 DATA 4,16,28,R2,00,4
28240 DATA 16,R,R,15,16,R
28250 DATA R,15,16,R3
28260 DATA R6,0-2
28270 DATA 9,16,21,R3
28280 DATA 4,16,20,R3
28290 DATA 9,16,21,R3
28300 DATA R6
28310 DATA 9,16,21,R3
28320 DATA 4,16,20,R3
28330 DATA [1
28340 DATA 9,16,21,R3
28350 DATA ]
28360 DATA *2
28370 DATA [2
28380 DATA 9,16,21,R3
28390 DATA fin
```

Módulo 3.3.2: Inicialización

Un módulo más complejo que lo usual, por la necesidad de recoger y procesar los datos de la melodía antes de ir a la parte del programa que es capaz de hacer que suene.

Módulo 3.3.2: Líneas 10000 - 10160

```
10000 REM*******************************
10010 REM Inicializacion
10020 REM*******************************
10030 CLS:LOCATE 11,13:PRINT "Espere por
 favor"
10040 DIM a$(200,2),p$(2),p(2),d(2),s%(2
,500,4)
10050 FOR c=0 TO 2
10060 LET p=0
10070 READ a$(p,c)
10080 a$(p,c)=LOWER$(a$(p,c))
10090 IF a$(p,c)<>"fin" THEN p=p+1:GOTO
10070
10100 p$(c)="S"
10110 NEXT c
10120 ocre=0
10130 nore=0
10140 db=20
10150 GOSUB 25000
10160 GOSUB 15000
```

Comentarios

Línea 10040: La complejidad de la tarea será recogida por la diversidad de matrices que se emplearán a lo largo del programa. Estas matrices serán explicadas conforme vayan siendo usadas.

Líneas 10050-10110: Los datos son leídos y transferidos a las tres columnas de la matriz A$. Esta matriz, tal y como se ha preparado aquí, puede guardar hasta tres conjuntos de 200 datos cada uno; los tres conjuntos se corresponden con las tres voces o canales del 464. Los datos son cargados inicialmente en la primera columna. Cuando se encuentra la palabra «fin», el bucle continúa leyendo los datos, pero comienza a colocarlos en la siguiente columna de la matriz. Observe, por tanto, que es vital terminar los datos correspondientes a cada voz con la palabra «fin» (tal y como se hace en los módulos DATA de muestra). Por último, a cada elemento de la matriz P$, correspondiente a cada voz, se le da el valor «S» para indicar (al menos inicialmente) que ese canal en particular debe sonar.

Línea 10120: La variable OCRE se utilizará para determinar en qué

octava estará basada la melodía; los datos de la melodía serán interpretados en relación a esta octava base.
Línea 10130: Similarmente, NORE es la nota base. Si se la cambia a lo largo del programa, cambiará la clave en la que está sonando la melodía.
Línea 10140: La variable DB se utiliza para almacenar la duración básica de una nota. Dentro de los datos de la melodía, cada nota será un múltiplo de la duración básica.
Líneas 10150-10160: Llaman a dos subrutinas que comienzan a transformar los datos de la melodía en una forma más digerible para el 464.

Comprobación

Necesitará introducir dos líneas provisionales:

```
15000 RETURN
25000 RETURN
```

Ahora lance el programa. Debería haber una pausa considerable antes de que el cursor vuelva a aparecer en pantalla. Aún no puede hacer nada con los datos que se han recogido, pero al menos ahora sabrá que no hay errores sintácticos en lo que ha introducido.

Módulo 3.3.3: La envolvente del sonido

La envolvente de una nota es básicamente el molde que dicta la forma en que la nota crece a partir del silencio, continúa sonando, y finalmente decrece hasta ser silencio de nuevo. La mayoría de los sonidos, y de los instrumentos musicales, tienen envolventes bastante características. Las órdenes de envolvente para el programa se ponen aquí, para recordarle que en muchos aspectos son parte de los datos de la melodía. El cambio de las envolventes alterará el sonido de lo interpretado casi tanto como el cambio de las notas.

No hablaremos aquí de los tecnicismos de las envolventes. Cualquier discusión útil nos llevaría muchas páginas y, sinceramente, puede aprender más experimentando con diferentes ajustes una vez se haya introducido todo el programa.

Módulo 3.3.3: Líneas 25000 - 25040

```
25000 REM******************************
25010 REM Poner aqui ordenes ENV y ENT
25020 REM******************************
25030 ENV 1,7,-1,10,8,-1,40
25040 RETURN
```

Módulo 3.3.4: Proceso de los datos de la melodía

A estas alturas, algunos estarán sorprendidos de que no vayamos a ponernos a tocar los datos de la melodía contenidos en las sentencias data. En realidad, aún queda un buen rato para eso, primero hay que procesar dichos datos.

La razón de ello es que, en nuestra búsqueda de la forma más fácil de registrar la melodía, nos hemos apartado del formato exigido por las diferentes órdenes de sonido. La notación utilizada en las sentencias data es fácil de recordar y utilizar, como comprobará cuando pasemos a analizar las distintas órdenes contenidas en ellas, pero tiene que ser transformada antes de que pueda ser introducida en la orden SOUND del 464. Rápido como es el BASIC del 464, la tarea de traducir las órdenes para hasta tres voces y de hacerlas sonar simultáneamente está fuera de su alcance. Podría hacerse, pero habría una limitación inaceptable en la velocidad a la que se podría tocar cualquier melodía.

Por consiguiente, el método adoptado es efectuar primero cualquier transformación necesaria, almacenar los parámetros de cada una de las notas que vayan a ser tocadas en la matriz S% y solamente entonces tocar la melodía, utilizando S% como fuente.

Módulo 3.3.4: Líneas 15000 - 15140

```
15000 REM******************************
15010 REM Compilar datos
15020 REM******************************
15030 FOR c=0 TO 2
15040 d=1:v=12:o=0:ev=0:et=0:p=0:r=1
15050 n=0
15060 WHILE a$(p,c)<>"fin"
15070 FOR i=1 TO 8
15080 IF LEFT$(a$(p,c),1)<>MID$("ovd*[]e
r",i,1) THEN NEXT i
15090 ON i GOSUB 16000,17000,18000,19000
,20000,21000,22000,23000,24000
15100 p=p+1
15110 WEND
15120 d(c)=n
15130 NEXT c
15140 RETURN
```

Comentarios

Líneas 15030-15130: El proceso se lleva a cabo para las tres voces, o canales, del 464.

Línea 15040: Las variables de esta línea se utilizarán para los objetivos siguientes:

D-duración de la nota
V-volumen
O-octava
EV-envolvente de volumen
ET-envolvente de tono
P-la posición en la secuencia de órdenes de una voz
N-el número de notas tocadas

Líneas 15070-15090: Similar a la técnica MENU$ utilizada en el programa Caracteres. Aquí el bucle compara la orden contenida en A$, con las órdenes permitidas registradas en la cadena de la línea 15080. Si la primera letra de la orden concuerda con una de las letras de la cadena, entonces la variable I del bucle registra la posición de la orden, posición que es utilizada por ON...GOSUB. El efecto de cada orden particular será examinado en los módulos que se ocupan de ellas. Si la primera letra de la orden *no* es ninguna de las letras de la cadena de órdenes, se supone que la orden es un valor de nota.
Línea 15120: El número de notas de cada voz se almacena en la matriz D.

Comprobación

El único test, y no es gran cosa, es colocar un RETURN en cada uno de los destinos especificados en la línea 15090. De esta forma, el programa correrá y usted podrá ver que no hay errores sintácticos. La adición de estas líneas con RETURN le permitirá, además, ejecutar el programa cada vez que introduzca uno de los módulos posteriores que realizan el proceso de los datos de la melodía.

Módulo 3.3.5: El valor de una nota

Naturalmente, la mayor parte de la melodía se va a componer de notas, y es este módulo el que coge las representaciones de las notas de los datos de la melodía y las hace comprensibles.

Los valores de las notas en sí se introducen basándose en el sistema de 12 tonos. En la música occidental, una octava se divide en siete tonos y cinco semitonos, 12 notas en total, por lo que cada nota de la octava puede representarse por un número comprendido entre 0 y 11. Puesto que hay una relación matemática clara entre las notas de una octava, o las notas de diferentes octavas, es perfectamente posible ampliar el intervalo 0 a 11, asignando el 12 a la primera

nota de la siguiente octava superior, el 24 a la primera de la octava que va después, y así sucesivamente. Este módulo se encargará de hacer el trabajo pesado de la transformación.

Módulo 3.3.5: Líneas 24000 - 24070

```
24000 REM******************************
24010 REM Nota
24020 REM******************************
24030 fr=440*(2^(o+(VAL(a$(p,c))-9)/12))
24040 tp=ROUND(125000/fr)
24050 s%(c,n,0)=tp:s%(c,n,1)=d:s%(c,n,2)
=v:s%(c,n,3)=ev:s%(c,n,4)=et
24060 n=n+1
24070 RETURN
```

Comentarios
Línea 24030: Esta fórmula transforma el valor de la nota (teniendo en cuenta el valor de la octava, 0, que discutiremos más tarde) en un valor de frecuencia.
Línea 24040: Habiendo llegado al valor de una frecuencia, esta línea lo transforma en un valor que se puede utilizar para accionar la orden SOUND; no hay nada de especial en el método empleado, es simplemente que el 464 está diseñado para trabajar con valores del tipo 125000/FRECUENCIA REAL.
Línea 24050: La presencia de un valor de nota en los datos de la melodía, se toma como una orden para tocar esa nota, así que los diversos elementos que conformarán una orden SOUND, son transferidos desde las variables donde están almacenados a una línea de la matriz S%, que será utilizada finalmente para tocar la melodía.

Módulo 3.3.6: Cambio de octava

Habrá observado que en la fórmula para calcular el valor de una nota se tiene en cuenta el valor de la octava (0). Esto da una mayor flexibilidad en la forma en que se pueden introducir las notas. La primera nota de la segunda octava puede introducirse con el valor 12, o se puede cambiar la octava al valor 1, y entonces el valor de la nota será 0. El método para especificar un cambio de octava es incluir una O seguida de un número.

Módulo 3.3.6: Líneas 16000 - 16040

```
16000 REM*******************************
16010 REM Octava
16020 REM*******************************
16030 o=VAL(MID$(a$(p,c),2))
16040 RETURN
```

Módulo 3.3.7: Especificación del volumen

El volumen se especifica incluyendo, en los datos de la melodía, una V seguida de un ajuste válido de volumen.

Módulo 3.3.7: Líneas 17000 - 17040

```
17000 REM*******************************
17010 REM Volumen
17020 REM*******************************
17030 v=VAL(MID$(a$(p,c),2))
17040 RETURN
```

Módulo 3.3.8: Ajuste de la duración de una nota

La duración de una nota se especifica incluyendo, en los datos de la melodía, una D seguida de un valor. El valor (duración) introducido será multiplicado por la duración básica de nota del módulo de inicialización.

Módulo 3.3.8: Líneas 18000 - 18040

```
18000 REM*******************************
18010 REM Duracion
18020 REM*******************************
18030 d=VAL(MID$(a$(p,c),2))
18040 RETURN
```

Módulo 3.3.9: Repetición de una sección de la melodía

La mayor parte de las melodías contienen secciones que se repiten, realmente una melodía sin repeticiones sería posiblemente un poco molesta de escuchar. A veces se repiten pequeñas secciones, pero otras veces se vuelven a tocar partes extensas de la melodía. Para ahorrar espacio (de memoria) tiene sentido incluir en el pro-

grama la posibilidad de marcar una sección que deba ser tocada dos veces seguidas. En este programa, a una sección como ésa se le pone un asterisco en su comienzo, y a su final un asterisco seguido por un número que indica las veces que debe ser interpretada dicha sección.

Módulo 3.3.9: Líneas 19000 - 19090

```
19000 REM*******************************
19010 REM Repetir
19020 REM*******************************
19030 IF a$(p,c)="*" THEN r=1:RETURN
19040 IF VAL(MID$(a$(p,c),2))=r THEN RET
URN
19050 r=r+1
19060 WHILE a$(p,c)<>"*"
19070 p=p-1:IF p=-1 THEN RETURN
19080 WEND
19090 RETURN
```

Comentarios
Línea 19030: Para acceder a este módulo, la orden contenida en los datos de la melodía debe *comenzar* con un asterisco. Si se trata de un asterisco solo, la variable R se hace igual a 1 (esta variable almacena el número de veces que se ha repetido la sección). El control se devuelve entonces a la parte principal del programa, para que pueda continuarse la traducción de la melodía.
Línea 19040: Si el asterisco va seguido de un número, significa que estamos al final de una sección. El valor de R, que muestra cuantas veces se ha repetido la sección, se compara con el número que va detrás del asterisco, que indica el número de veces que *debería* haber sido repetida. Si ambos números son iguales, entonces el programa deja la sección y continúa con lo que va detrás de ella.
Líneas 19050-19080: Se incrementa el número de repeticiones registrado y el programa retrocede, antes de volver a la parte principal, al comienzo de la sección que se ha de repetir.

Módulo 3.3.10: Variaciones en las secciones que se repiten

Es muy frecuente el caso en que, al repetir una sección, se hacen ligeros cambios en las notas; la segunda repetición puede culminar en una serie de notas altas, en tanto que la primera terminaba con notas más bajas. Este módulo le permite al usuario especificar que, en una sección que se ha de repetir, un cierto número de notas se

deben incluir solamente durante *una* de las repeticiones. De esta forma, por ejemplo, se podría repetir dos veces una sección, con unas notas finales en la primera repetición y otras en la segunda. El método empleado en este programa, es marcar el comienzo de la subsección con un símbolo [seguido de un número que representa la repetición en la que se tocará la subsección, y marcando el final de la subsección con un único símbolo de].

Módulo 3.3.10: Líneas 20000 - 21030

```
20000 REM*******************************
20010 REM En repeticion n-sima hacer
20020 REM*******************************
20030 IF VAL(MID$(a$(p,c),2))=r THEN RET
URN
20040 WHILE a$(p,c)<>"]" AND a$(p+1,c)<>
"fin"
20050 p=p+1
20060 WEND
20070 RETURN
21000 REM*******************************
21010 REM Terminar de hacer
21020 REM*******************************
21030 RETURN
```

Comentarios
Línea 20030: Para llegar a este punto, el programa debe haber encontrado un símbolo [. La línea examina el número que va detrás de dicho símbolo, y si el número es igual al valor de R (que registra el número de veces que se ha repetido la sección en cuestión), entonces se le permite al programa continuar tocando las notas que van detrás de].
Líneas 20040-20060: Este bucle se utiliza si la subsección no se va a tocar durante la repetición presente, y lo que hace simplemente es explorar más allá del comienzo de la subsección hasta encontrar el símbolo] de terminación.
Línea 21030: Cuando se encuentra un], lo que esta línea hace es permitir simplemente que el programa continúe.

Módulo 3.3.11: Cambio de las envolventes

Ya hemos preparado una envolvente, pero sabrá que el 464 es capaz de definir hasta 16 envolventes para el volumen y el tono. Incluyendo ENV o ENT seguido (sin espacios en blanco) de un número

válido para una envolvente que haya definido en 25000, se cambia la envolvente utilizada por la orden SOUND.

Módulo 3.3.11: Líneas 22000 - 22050

```
22000 REM*******************************
22010 REM Envolventes
22020 REM*******************************
22030 IF LEFT$(a$(p,c),3)="env" THEN ev=
VAL(MID$(a$(p,c),4))
22040 IF LEFT$(a$(p,c),3)="ent" THEN et=
VAL(MID$(a$(p,c),4))
22050 RETURN
```

Módulo 3.3.12: Generación de un silencio

No debe olvidarse que en una melodía, un silencio es tan importante como una nota, así que necesitamos atender la posibilidad de parar el sonido durante un momento. En este programa, se crea un silencio mediante la inclusión de una R en los datos de la melodía. Una única R, genera una pausa de duración igual a la duración básica de las notas, en tanto que R seguida de un número, genera una pausa de duración igual a la duración correspondiente a ese número.

Módulo 3.3.12: Líneas 23000 - 23060

```
23000 REM*******************************
23010 REM Pausa
23020 REM*******************************
23030 IF a$(p,c)="r" THEN du=d ELSE du=V
AL(MID$(a$(p,c),2))*d
23040 s%(c,n,0)=0:s%(c,n,1)=du:s%(c,n,2)
=0:s%(c,n,3)=0:s%(c,n,4)=0
23050 n=n+1
23060 RETURN
```

Comprobación

Si no lo ha ido haciendo al introducir cada uno de los últimos módulos, ahora puede lanzar el programa. Cada una de las secciones irá siendo llamada conforme se analizan los datos de la melodía.

Módulo 3.3.13: El menú del programa

Un menú sencillo que permite acceder al resto de las funciones del programa.

Módulo 3.3.13: Líneas 11000 - 11140

```
11000 REM*******************************
11010 REM Control
11020 REM*******************************
11030 WHILE o<>3
11040 CLS:PRINT TAB(17);"MUSICA";TAB(17)
;"======"
11050 PRINT:PRINT "Opciones:":PRINT
11060 PRINT "1) Cambiar parametros"
11070 PRINT "2) Tocar musica"
11080 PRINT "3) Terminar"
11090 PRINT:INPUT "Introduzca opcion des
eada: ",o
11100 ON o GOSUB 12000,13000
11110 WEND
11120 CLS
11130 LIST 25000-
11140 END
```

Módulo 3.3.14: Cambio de los parámetros de la melodía

Cuando se pasa a tocar la melodía, se usarán varios parámetros que el usuario puede modificar utilizando este módulo.

Módulo 3.3.14: Líneas 12000 - 12110

```
12000 REM*******************************
12010 REM Cambiar parametros
12020 REM*******************************
12030 CLS:PRINT TAB(12);"CAMBIAR PARAMET
ROS";TAB(12);"==================":PRINT
12040 PEN 2:PRINT "Pulse ENTER";:PEN 1:P
RINT " para dejar parametro inalterado":
PRINT
12050 PRINT "Octava relativa (";ocre;:IN
PUT") :",x$:IF x$<>"" THEN ocre=VAL(x$)
12060 PRINT "Nota relativa (";nore;:INPU
T") :",x$:IF x$<>"" THEN nore=VAL(x$)
12070 PRINT "Duracion unitaria (";db;:IN
PUT") :",x$:IF x$<>"" THEN db=VAL(x$)
12080 FOR c=0 TO 2
12090 PRINT "Tocar canal";c;"(";p$(c);:I
NPUT") :",x$:IF x$<>"" THEN p$(c)=UPPER$
(x$)
12100 NEXT c
12110 RETURN
```

Comentarios

Línea 12050: La octava relativa es la base a partir de la cual se calcula la nota que se va a tocar.

Línea 12060: Teniendo la octava base, las notas a tocar se pueden calcular en base a una nota particular de esa octava, cambiándose por tanto la clave de la interpretación musical.

Línea 12070: La duración básica, o unitaria, que se utiliza cuando se toca una nota. Cuanto mayor sea el valor introducido, más lenta será la interpretación.

Línea12080-12100: No todas las voces del 464 tienen por qué sonar; este bucle permite excluir cualquiera de las voces.

Comprobación

Si se lanza el programa, tras una pausa para recalcular los datos de la melodía, debería aparecer el menú y debería poder seleccionar la opción 1 para especificar los valores que desea tomen los parámetros.

Módulo 3.3.15: Control de la interpretación

Pasamos ahora a los dos módulos que se ocuparán del objetivo principal del programa: tocar una melodía. El primero de ellos es un módulo de control que asigna trabajos y atiende al final de la melodía. El siguiente módulo hará el trabajo de tocar la melodía en sí.

Módulo 3.3.15: Líneas 13000 - 13160

```
13000 REM******************************
13010 REM Tocar musica
13020 REM******************************
13030 rtp=2^(ocre+nore/12)
13040 FOR c=0 TO 2
13050 p(c)=0
13060 NEXT c
13070 SOUND 199,0,0
13080 GOSUB 14000
13090 GOSUB 14000
13100 GOSUB 14000
13110 RELEASE 7
13120 hecho=0
13130 WHILE NOT hecho
13140 GOSUB 14000
13150 WEND
13160 RETURN
```

Comentarios
Línea 13030: La base a partir de la cual se tocará la melodía.
Líneas 13040-13060: La matriz P de cuenta, de las tres voces, se pone a 0.
Línea 13070: Esta línea no envía una nota, se trata de una orden de preparación que hace tres cosas:

1) Limpia cualesquiera notas que puedan estar en cola, pertenecientes a algún uso anterior, y esperando ser tocadas.
2) Especifica qué órdenes se van a enviar a las tres voces.
3) Coloca un calderón (suspensión de la medida del compás), de forma que no se puede tocar ninguna nota.

El número 199 carece de significado en sí mismo: se trata simplemente de una forma de poner ciertos dígitos binarios o «bits» a 1. En este caso concreto, se activan los bits 0, 1, 2, 6 y 7. Los pormenores de los efectos que producen estos bits se dan en el Capítulo 6 de su manual.
Líneas 13080-13110: Estas tres llamadas al módulo siguiente preparan una cola de notas a tocar. Esto significa que cualquier pequeño retardo, introducido por el bucle que interpreta la parte principal de la melodía no tendrá ningún efecto en la regularidad o ritmo de la interpretación. La orden RELEASE de la línea 13110 hace que las tres voces comiencen simultáneamente, una vez que se han formado las primeras notas de la cola. Esta capacidad de hacer colas es una de las características musicales más potentes del 464. En casi todos los otros micros domésticos, el problema de sincronizar la interpretación de las notas es extremadamente difícil; un pequeño retardo en el proceso de una nota dará lugar a una sincronización irregular o desigual. Con el 464 ese problema está eliminado y el programa puede ser más sencillo así como más eficaz.
Líneas 13120-13140: El resto de proceso de tocar la melodía, es simplemente una cuestión de llamar continuamente al módulo siguiente, hasta que la variable HECHO (que es activada por ese otro módulo) indique que se han agotado los datos para la melodía.

Módulo 3.3.16: Interpretación de las notas

El módulo final que hace que suenen las notas de las tres voces.

Módulo 3.3.16: Líneas 14000 - 14120

```
14000 REM*****************************
14010 REM Explorar canales
14020 REM*****************************
```

```
14030 hecho=-1
14040 FOR c=0 TO 2
14050 IF p(c)=d(c) OR p$(c)="N" THEN GOT
O 14110
14060 hecho=0
14070 IF (SQ(2^c) AND 7)=0 THEN GOTO 141
10
14080 n=p(c)
14090 SOUND 2^c,s%(c,n,0)*rtp,s%(c,n,1)*
db,s%(c,n,2),s%(c,n,3),s%(c,n,4)
14100 p(c)=p(c)+1
14110 NEXT c
14120 RETURN
```

Comentarios

Líneas 14040-14050 y 14110: Independientemente de si van a sonar o no las tres voces, este bucle está preparado para explorarlas a las tres. Sin embargo, la línea 14050 detecta si una voz en particular ha agotado todos sus datos, o si el usuario ha reajustado los parámetros para excluir una determinada voz. Si se accede a la parte principal del bucle (es decir: aún hay notas a tocar, al menos en una de las voces), entonces el valor de HECHO se pone a 0.

Línea 14070: La función SQ se utiliza para determinar si aún es posible poner una nota en la cola de la voz en particular.

Líneas 14080-14100: Si *es* posible añadir otra nota a la cola, el siguiente conjunto de datos de S% se utiliza como base para una orden SOUND; observe que esto no hace que suene inmediatamente una nota, simplemente la añade a la cola de notas que están esperando a ser tocadas. Por último se incrementa la cuenta de la voz en particular.

Comprobación

Al fin puede hacer una comprobación significativa del programa. Para ello sólo necesita lanzarlo y, cuando aparezca el menú, especificar la opción 2. ¡Sabrá si funciona o no!

4. Más y más serio

En esta fase de sus progresos debería estar más familiarizado con las posibilidades de su 464 y con algunas de las técnicas necesarias para ponerlo a trabajar para Vd. Por tanto, ha llegado el momento de considerar algunos programas de mayor entidad, que le permitirán hacer a su 464 lo que mejor hacen los microordenadores: manejar, clasificar y recuperar información en beneficio de sus propietarios.

Los programas que se incluyen en este capítulo son:

UNIARCHIVO: Un potente sistema de fichero personal capaz de almacenar una amplia variedad de información, para su inmediata recuperación.

NNUMERO: Un programa que crea un diccionario de Nombres y Números para utilizar en casi cualquier asunto que desee. Permite al usuario la expedición de facturas, la estimación de existencias o incluso la cuenta de las calorías del menú del día.

EDITEXTO: Un sencillo paquete de tratamiento de textos que corre bajo BASIC.

MULTIRRESPUESTA: Un generador de preguntas a las que se le asocian varias respuestas, de las cuales sólo una es la correcta.

PROGRAMA 4.1: Uniarchivo

Función del programa

Uniarchivo es una preciosidad de programa que se ha ido desarrollando, a lo largo de los años, en los libros «Programas prácticos para el...». Los lectores de libros anteriores me han escrito para decirme que lo están utilizando en su negocio, para enseñar a los alum-

nos la forma en que los micros pueden manejar la información, para ayudar a clubs y asociaciones benéficas, o simplemente para seguir la pista de los libros y los discos de casa.

Entre los nuevos conceptos introducidos en este programa se incluyen:

1) Búsqueda binaria.
2) Empaquetamiento de unidades de información en cadenas continuas.
3) Comienzo de un programa sin borrar matrices.

Módulo 4.1.1: Preparación de la estructura del fichero

Muchos libros dirigidos a los propietarios de micros domésticos presentan programas clasificadores (archivadores) de calidad inferior, que además son de utilización extremadamente poco flexible; los programas están construidos de tal forma, que cada vez que el usuario almacena algo, tiene que ser bajo encabezamientos tales como, digamos: Nombre, Dirección, N.° Teléfono, o de estructura similar. La belleza de Uniarchivo es que, en tanto que (por supuesto) le permite utilizar tal estructura, también le permitirá crear otros ficheros de estructura muy diferente (quizá con un único cabecero, quizá con diez) sin tener que hacer ningún cambio en el propio programa. Uniarchivo

```
ENTRADA 1 :

APELLIDO: LANDABURU
NOMBRE: JAVIER
DIRECCION 1:UNA CALLE 11
DIRECCION 2:UN PUEBLO
DIRECCION 3:UNA CIUDAD
TELEFONO:01 234 5678

ORDENES DISPONIBLES:

ENTER para ver el elemento siguiente.
'R' para rectificar.
'C' para continuar busqueda.
'#' seguido de un numero para mover pun-
    tero.
'\' para abandonar busqueda.

Cual elige:
```

Fig. 4.1 Uniarchivo en modo de búsqueda.

es de los que me gusta llamar programas camaleón: programas que se adaptan a una gran diversidad de usos diferentes, reaccionando ante el usuario en formas distintas que dependen de la tarea que está realizando en el momento.

El objetivo de este primer módulo es inicializar algunas variables, y lo que es más importante: permitir al usuario que prepare su fichero particular, exactamente en la forma deseada.

Módulo 4.1.1: Líneas 20000 - 20110

```
20000 REM*******************************
20010 REM Crear fichero
20020 REM*******************************
20030 CLS:PRINT TAB(13);"FICHERO NUEVO":
PRINT TAB(13);"=============="
20040 PRINT:INPUT "Esta seguro (s/n)";r$
:IF LEFT$(UPPER$(r$),1)="N" THEN RETURN
20050 CLEAR:DIM lista$(1000)
20060 PRINT:INPUT "Cuantos elementos por
 cada entrada";x
20070 DIM elemento$(x-1),ptr(x-1)
20080 PRINT:FOR i=0 TO x-1
20090 PRINT "Denominacion del elemento #
";i+1;:INPUT":",elemento$(i)
20095 elemento$(i)=UPPER$(elemento$(i))
20100 NEXT i
20110 in=-1:GOTO 11000
```

Comentarios

Líneas 20030-20040: La creación de ficheros nuevos borrará cualquier información que haya en memoria en ese momento, así que se le da al usuario la oportunidad de detenerse en este punto.

Línea 20050: La información a guardar en Uniarchivo se almacenará en la matriz LISTA$. En esta línea, la matriz es dimensionada para permitir 1001 entradas en el fichero. Si lo desea puede incrementar el número a 2000 o más. Lo único que hay que tener presente es que cada elemento de la matriz, antes de ser utilizado, ocupa tres octetos de memoria. Si dimensiona la matriz a 5000 elementos ocupará hasta 15000 octetos de memoria, la tercera parte de la memoria libre, y todo ello antes de que haya almacenado información alguna. Es una simple cuestión de apreciación: si cree que necesitará menos de 1000 entradas en un fichero, entonces puede también rebajar el número de la sentencia DIM y ahorrar un poco de memoria. Observe también que, dado que estamos utilizando un elemento de una matriz para almacenar cada entrada, la máxima longitud que puede tener la en-

trada (incluyendo sus caracteres separadores) es de 255 caracteres: la longitud máxima de cadena que puede manejar el 464.
Líneas 20060-20100: Parte del secreto de la flexibilidad de Uniarchivo. Por cada *entrada* (a la que para ayudarse puede imaginarla como si fuera una ficha) puede definir el número de elementos que aparecerán en ella. Si estuviera clasificando su colección de discos, podría pensar en encabezamientos como los siguientes: SURCO, ALBUM, COMPOSITOR, ARTISTA, DURACION. En este caso especificaría cinco elementos, y entonces introduciría la denominación de cada uno de ellos. En el futuro, cada vez que utilice Uniarchivo para almacenar u obtener información sobre su colección de discos, se le pedirá que ponga un dato bajo cada uno de esos encabezamientos. Para las personas con mente más técnica: aunque llamaremos *entrada* a toda la ficha y *elementos* a los encabezamientos individuales, en la jerga informática se les denominaría como *registros* y *campos* respectivamente. El uso de las variables que se definen aquí, se explicará a lo largo de los comentarios del programa.

Módulo 4.1.2: Arranque del programa

La posibilidad de crear un nuevo fichero no se utilizará en cada ocasión en que se ejecute el programa. En la mayoría de las ocasiones, lo que el usuario deseará es cargar datos desde la cinta. Este módulo de arranque simplemente da al usuario la oportunidad de decir cuál de las dos opciones desea.

Módulo 4.1.2: Líneas 10000 - 10150

```
10000 REM*******************************
10010 REM Comenzar
10020 REM*******************************
10030 WHILE NOT in
10040 CLS
10050 PRINT TAB(15);"UNIARCHIVO"
10060 PRINT TAB(15);"==========="
10070 PRINT
10080 PRINT "ORDENES DISPONIBLES:"
10090 PRINT
10100 PRINT "1) Crear fichero nuevo"
10110 PRINT "2) Cargar fichero desde la
cinta"
10120 PRINT
10130 INPUT "Cual elige: ",z
10140 ON z GOSUB 20000,21000
10150 WEND
```

Comentarios
Líneas 10030 y 10150: Estas dos líneas representan una técnica muy útil que indudablemente deseará utilizar en sus propios programas. Cuando el programa es inicializado por el primer módulo que introdujo, se le dio el valor –1 a una variable denominada IN (abreviación de INicializado). La razón para hacer esto queda clara en este bucle: si cuando se llega a la línea 10030 el valor de IN es –1, entonces no se ejecutará nada de lo que hay en el bucle. El operador NOT que hay detrás de WHILE, en la línea 10030, asegura que sólo se entrará en el bucle si el valor de IN es 0. Lo que esto significa es que, si detiene el programa y lo arranca de nuevo con un GOTO 10000 (o un GOTO 1 si ha incluido el pequeño módulo SAVE que recomendé con el primer programa del libro), no se perderá ninguno de los datos de la memoria.

Comprobación
Introduzca una línea provisional:

```
11000 STOP
```

Lance el programa y especifique que desea preparar un fichero nuevo. Siga las peticiones que se le hacen e introduzca algunas cifras y nombres lógicos en respuesta. El programa debería detenerse entonces. Ahora introduzca:

```
GOTO 10000[ENTER]
```

y el programa debería detenerse inmediatamente: ha detectado que hay un fichero en memoria, aún cuando está vacío, y se ha saltado todo el módulo que comienza en la línea 10000.

Módulo 4.1.3: El menú

Un menú estándar, pero observe que no hay medidas para finalizar el programa por medio de ON ERROR y la tecla ESC. Uniarchivo es un programa complejo, y pararlo a mitad de flujo podría acarrear el deterioro de la información que guarda, por haber quedado incompleto algún proceso importante. Para finalizar Uniarchivo *siempre* debería volver a este menú y usar la opción 6.

Módulo 4.1.3: Líneas 11000 - 11210

```
11000 REM*******************************
11010 REM Menu principal
11020 REM*******************************
```

```
11030 WHILE NOT hecho
11040 CLS
11050 PRINT TAB(15);"UNIARCHIVO"
11060 PRINT TAB(15);"==========="
11070 PRINT
11080 PRINT "ORDENES DISPONIBLES:"
11090 PRINT
11100 PRINT "1) Crear fichero nuevo"
11110 PRINT "2) Cargar fichero desde la
cinta"
11120 PRINT "3) Introducir informacion"
11130 PRINT "4) Buscar/Mostrar/Cambiar"
11140 PRINT "5) Grabar fichero en cinta"

11150 PRINT "6) Terminar"
11160 PRINT
11170 INPUT "Cual elige: ",z
11180 ON z GOSUB 20000,21000,12000,17000
,18000,22000
11190 WEND
11200 hecho=0
11210 END
```

Módulo 4.1.4: Parada del programa

Este pequeño módulo muestra el mensaje de finalización del programa, y activa el valor de la variable HECHO para informar al menú que el programa se ha terminado.

Módulo 4.1.4: Líneas 22000 - 22060

```
22000 REM*****************************
22010 REM Terminar
22020 REM*****************************
22030 hecho=-1
22040 CLS:LOCATE 11,13
22050 PRINT "FICHERO CERRADO":PRINT:PRIN
T
22060 RETURN
25000 OPENIN "uniarchivo"
25010 INPUT t$
25020 PRINT t$
25030 GOTO 25010
```

Comprobación

Hasta podría teclear:

GOTO 11000[ENTER]

y ver el menú presentado en pantalla. Especifique entonces la opción 6 y vea el programa finalizado.

Módulo 4.1.5: Grabación de datos en cinta

Conforme comience a desarrollar programas más complejos, bien sea a partir de este libro o bien solo, encontrará más y más deseable introducir el módulo de FICHERO DE DATOS lo antes posible. En este caso en concreto (y en general), la razón para ello es que solamente es posible hacer comprobaciones adecuadas de Uniarchivo (del programa) introduciendo cantidades bastante considerables de datos. Puesto que está destinado a cometer equivocaciones que pueden deteriorar los datos, se enfrenta a la perspectiva de tener que reintroducirlos una y otra vez, conforme vaya desarrollando el programa. La solución es almacenarlos en cinta desde las primeras etapas, llamando entonces a dicha información para hacer sus comprobaciones.

Este módulo le permitirá almacenar los datos, y le presentará una idea que hemos mencionado antes pero que aún no hemos utilizado: dar a un programa la posibilidad de coger o almacenar ficheros con una amplia gama de nombres.

Módulo 4.1.5: Líneas 18000 - 18080

```
18000 REM*******************************
18010 REM Grabar fichero
18020 REM*******************************
18030 CLS:PRINT TAB(13);"GRABAR FICHERO"
:PRINT TAB(13);"=============="
18040 PRINT:INPUT "Nombre bajo el que de
sea grabarlo:       ",fi$
18050 PRINT:OPENOUT fi$:PRINT #9,el:PRIN
T #9,x
18060 FOR i=0 TO el-1:PRINT #9,lista$(i)
:NEXT i
18070 FOR i=0 TO x-1:PRINT #9,elemento$(
i):NEXT i
18080 CLOSEOUT:RETURN
```

Comentarios
Líneas 18030-18040: Se especifica interactivamente el nombre del fichero.
Líneas 18050-18070: El número de entradas guardado por Uniarchivo en cualquier momento está almacenado en la variable EL, en tanto que X registra el número de elementos en cada entrada. Estas líneas graban todos los elementos útiles de LISTA$ y los encabezamientos de ELEMENTO$.

Módulo 4.1.6: Carga de datos desde la cinta

Habiendo puesto los datos en la cinta, el trabajo que hay que hacer ahora es recuperarlos. No se trata de una tarea tan simple ya que, así como cuando los datos fueron grabados todas las matrices estaban dimensionadas correctamente en memoria, cuando un fichero se carga desde la cinta todas las matrices tienen que ser preparadas de nuevo. Esta es la razón por la que se colocaron las variables EL y X al comienzo del fichero, de forma que pueden ser leídas e introducidas en memoria en primer lugar, y utilizadas entonces para dimensionar las matrices justamente en la misma forma del primer módulo, en el que fueron introducidas por el usuario en vez de por el DATACORDER.

Módulo 4.1.6: Líneas 21000 - 21140

```
21000 REM*******************************
21010 REM Cargar fichero
21020 REM*******************************
21030 CLS:PRINT TAB(13);"CARGAR FICHERO"
:PRINT TAB(13);"==============="
21040 PRINT:INPUT "Esta seguro (s/n)";r$
:IF LEFT$(UPPER$(r$),1)="N" THEN RETURN
21050 CLEAR:DIM lista$(1000)
21060 PRINT:INPUT "Nombre del fichero a
cargar: ",fi$
21070 PRINT:OPENIN fi$:INPUT #9,el,x:DIM
 elemento$(x-1),ptr(x-1)
21080 FOR i=0 TO el-1
21090 INPUT #9,lista$(i)
21100 NEXT i
21110 FOR i=0 TO x-1
21120 INPUT #9,elemento$(i)
21130 NEXT i
21140 in=-1:CLOSEIN:GOTO 11000
```

Comentarios

Línea 21140: Muchas personas pueden quedarse sorprendidas por la presencia de un GOTO, aquí, al final de la subrutina. Este GOTO no puede ser sustituido por un RETURN, ya que en el transcurso del módulo se ha limpiado la memoria para permitir el redimensionado de las matrices. Al borrar la memoria se perdió todo registro de la orden GOSUB original, por lo que un RETURN únicamente generaría un mensaje de error.

Comprobación

Lance el programa y especifique que desea crear un fichero nuevo. Prepare cuatro elementos de nombres UNO, DOS, etc. Cuando salga el menú, escoja la opción 5 y entonces especifique un nombre de fichero tal como UNIDATOS. Antes de comenzar a grabar asegúrese de que hay una cinta adecuada en el Datacorder. Cuando el programa retorne al menú, párelo con la opción 6 y ejecútelo de nuevo. Esta vez especifique que desea cargar desde la cinta, especifique el nombre del fichero de datos y, cuando se le dé la orden, introduzca el fichero de datos poniendo la cinta en marcha (habiéndola rebobinado previamente, claro está). Cuando aparezca el menú, pare el programa de nuevo y teclee:

```
FOR I=0 TO X-1:PRINT ELEMENTO$(I):NEXT[ENTER]
```

debiendo ser el resultado la presentación de sus cuatro encabezamientos.

Módulo 4.1.7: Una forma mejor de búsqueda

En este módulo, y en los dos que siguen, echaremos un vistazo a la forma en que se añade una nueva entrada al fichero principal contenido en LISTA$. No obstante, la esencia del método se encuentra en éste, ya que es el módulo que permite a Uniarchivo buscar rápidamente a través de un gran fichero de entradas hasta encontrar la posición correcta en la que insertar una entrada nueva, o realizar una búsqueda rápida para localizar la presencia de una cierta entrada clave en el fichero.

El método es conocido como «búsqueda binaria», y se puede utilizar para reducir extraordinariamente el tiempo de búsqueda en cualquier programa que contenga grandes listas de datos ordenados. Considere el ejemplo siguiente.

Se ha creado un fichero que contiene 2000 nombres, y hay que insertar uno nuevo según el orden alfabético apropiado. Si hacemos

trampa y examinamos la lista de nombres podemos determinar la posición (digamos la 1731) en que debe ir el nuevo nombre (digamos «SUSANA»), aunque el ordenador no tiene forma de conocer esto de antemano.

Una cosa que podemos hacer es poner al ordenador a examinar nombre tras nombre desde el primero de ellos. Así que comenzará por «ABRAHAN» y se dará cuenta de que «SUSANA» debe ir después, entonces irá a «ADELA» y así sucesivamente. En determinado momento, tras examinar 1732 nombres, la búsqueda estará sobre un nombre tal como «TEODORO» que debe ir *después* de «SUSANA», con lo que se ha localizado la posición correcta.

Se trata de un método seguro, pero qué bueno sería si se pudiera reducir un poco el número de comparaciones a hacer. Bueno, pues en el caso de nuestro fichero de 2000 nombres, todo el proceso se puede realizar con *10 comparaciones* únicamente. He aquí la forma de conseguirlo:

El ordenador comienza la búsqueda examinando el nombre de la posición 1024 del fichero (se escoge 1024 porque es la mayor potencia de 2, en concreto «2^10», que está por debajo del número total de entradas: 2000). El ordenador encontrará que el nombre de la posición 1024 es anterior a «SUSANA» según orden alfabético (recuerde que ocupaba la posición 1731), por lo que suma 1024/2, o lo que es lo mismo 512, o lo que es lo mismo 2^9, a la posición primera de búsqueda, llegando pues a la 1536. Una vez más, el nombre que ocupa esta posición es anterior a «SUSANA», por lo que esta vez se suma 256, o 2^8, a 1536, dando lugar a 1792. Ahora ocurre algo diferente, ya que el nombre de la posición 1792 es *posterior* a «SUSANA», así que en vez de sumar 128, o 2^7, lo que se hace ahora es *restarlo* de la posición de búsqueda, obteniéndose la posición 1664.

La búsqueda continúa, sumándose o restándose potencias decrecientes de 2, según una pauta similar a la siguiente:

N.º DE LA COMPARACION	POSICION	ACCION
1	1024	+512
2	1536	+256
3	1792	–128
4	1664	+64
5	1728	+32
6	1760	–16
7	1744	–8
8	1736	–4
9	1732	–2
10	1730	+1

Pruebe con un número de entradas diferente y con diferentes posicones de búsqueda, verá como siempre funciona.

Módulo 4.1.7: Líneas 13000 - 13110

```
13000 REM*******************************
13010 REM Busqueda binaria
13020 REM*******************************
13030 IF el=0 THEN bb=0:RETURN
13040 po=INT(LOG(el)/LOG(2)):bb=2^po-1
13050 FOR i=po-1 TO 0 STEP -1
13060 bb=bb+2^i*((lista$(bb)>te$)-(lista
$(bb)<te$))
13070 IF bb<0 THEN bb=0
13080 IF bb>el-1 THEN bb=el-1
13090 NEXT i
13100 IF lista$(bb)<te$ THEN bb=bb+1
13110 RETURN
```

Comentarios
Línea 13030: Si no hubiera ninguna entrada en el fichero, los cálculos nos llevarían a un error. Esta línea evita que se produzca esta situación.
Línea 13040: Estas dos expresiones encuentran la mayor potencia de 2 que queda por debajo del número de elementos del fichero, e igualan entonces el puntero de búsqueda (BB) a ese número. El «–1» de la segunda expresión tiene en cuenta el hecho de que la matriz está numerada a partir de 0, no de 1.
Líneas 13050-13090: Este bucle dirige la búsqueda, utilizando potencias decrecientes de 2. El fichero principal está contenido en LISTA$, y la nueva entrada en TE$. El valor del puntero se fija mediante dos condiciones lógicas que indican si TE$ es mayor o menor que la entrada que hay en LISTA$(BB).
Línea 13100: En algunos casos se llega a una posición que queda a 1 posición por debajo de la posición correcta, en este caso BB se incrementa en 1.

Módulo 4.1.8: Inserción de una entrada

Este módulo inserta la nueva entrada en la posición indicada por la variable BB. Se hace simplemente moviendo todo una posición hacia arriba, desde BB en adelante.

Módulo 4.1.8: Líneas 14000 - 14070

```
14000 REM*******************************
14010 REM Insertar entrada
14020 REM*******************************
14030 IF el=0 THEN GOTO 14070
14040 FOR i=el TO bb+1 STEP -1
14050 lista$(i)=lista$(i-1)
14060 NEXT i
14070 lista$(bb)=te$:el=el+1:RETURN
```

Módulo 4.1.9: Haciendo entradas en el fichero

Habiendo introducido los módulos que hacen el trabajo real, podemos pasar ahora a aquel con el que se relacionará el usuario cuando introduzca material en el archivador. La función del módulo es guiar al usuario para que introduzca el número correcto de elementos (y en el orden correcto) por cada entrada, combinar estos elementos en una única cadena que se meterá en una línea de LISTA$, y llamar entonces a los dos módulos anteriores para insertar la entrada en el fichero principal.

Módulo 4.1.9: Líneas 12000 - 12320

```
12000 REM*******************************
12010 REM Nuevas entradas
12020 REM*******************************
12030 WHILE el<1000
12040 te$=""
12050 CLS
12060 PRINT TAB(13);"NUEVAS ENTRADAS"
12070 PRINT TAB(13);"================"
12080 PRINT
12090 PRINT "Numero de la entrada: ";el+
1
12100 PRINT
12110 PRINT "ORDENES DISPONIBLES:"
12120 PRINT
12130 PRINT "- Introducir elemento espec
ificado.":PRINT
12140 PRINT "- '\' para volver al menu p
rincipal."
12150 PRINT
12220 FOR i=0 TO x-1
12230 PRINT elemento$(i);:INPUT":",q$
```

```
12240 IF q$="\" THEN RETURN
12250 te$=te$+UPPER$(q$)+"\"
12260 NEXT i
12270 PRINT:PRINT "Por favor espere un m
omento ..."
12280 GOSUB 13000:GOSUB 14000
12290 WEND
12300 CLS
12310 PRINT "LO SIENTO, NO SON POSIBLES
MAS ENTRADAS  "
12320 FOR i=1 TO 2000:NEXT i:RETURN
```

Comentarios

Líneas 12030 y 12290-12320: Las entradas solamente son aceptadas en tanto quede sitio para ellas, en caso de que no quede sitio se genera un mensaje de error. Observe que si desea cambiar el número máximo de entradas, necesitará cambiar esta referencia a 1000 por el valor adecuado (podría reemplazarse por una variable, siempre que recuerde definir esa variable cuando se inicialice el programa, y que recuerde grabarla cuando se almacene en cinta un fichero particular).
Líneas 12220-12260: Estas son las líneas que solicitan la introducción de los elementos individuales de la nueva entrada. La denominación de cada elemento se toma de la matriz ELEMENTO$, que se preparó en el módulo de inicialización. Cada elemento introducido es guardado en Q$, finalizándose la introducción de nuevos elementos si Q$ llega a ser «\» en cualquier momento. Si lo que se introduce no es «\», el elemento es añadido a TE$ (que llegará a contener la totalidad de la entrada), y se añade un carácter «\» al final del elemento. El propósito de este último carácter «\» no tiene absolutamente nada que ver con su función anterior (que era dar por terminada la introducción de la información); en este caso se utiliza meramente como un indicador para que partes posteriores del programa puedan reconocer dónde acaba un cierto elemento y comienza el siguiente. Así una entrada tal como:

LANDABURU
JAVIER
UNA CALLE 11
UNA CIUDAD

se almacenará en la matriz LISTA$ como:

LANDABURU\JAVIER\UNA CALLE 11\UNA CIUDAD\

creándose lo que se conoce como una «cadena empaquetada». La razón para escoger «\» es que resulta bastante improbable que haya

algún motivo importante por el que alguien desee incluirlo en una entrada; si por cualquier razón tuviera que hacerlo, entonces escoja otro carácter separador. Observe que todos los elementos introducidos son transformados en mayúsculas; posteriormente verá que ocurre lo mismo cuando se buscan elementos en el fichero. Esto impide que la introducción inadvertida en letras mayúsculas o minúsculas haga difícil encontrar un elemento del fichero. Si *debe* utilizar mayúsculas y minúsculas conjuntamente, entonces puede suprimir estas medidas (conversión de minúsculas en mayúsculas) sin perjudicar el funcionamiento del programa; en ese caso, debe darse cuenta de que el 464 considera a cualquier minúscula como posterior (según orden alfabético) a cualquier mayúscula, lo que puede hacer que el orden de su fichero parezca un tanto singular.

Comprobación

Ahora está en situación de comprobar los tres últimos módulos que ha introducido. Lance el programa y prepare un fichero con dos elementos por entrada, llamados «UNO» y «DOS». Especifique la opción 3 una vez haya terminado la inicialización del programa y aparezca el menú principal. Se encontrará con la presentación creada por el módulo que acabamos de introducir, en la que se le pedirá que introduzca el elemento «UNO». En respuesta, introduzca «AA1». La petición se repetirá para el elemento «DOS», a lo que debe responder con «AA2». Repita el proceso contestando «DD1», «DD2», «CC1», «CC2», «BB1» y «BB2» a las peticiones siguientes. Ahora meta «\» y volverá al menú principal. Seleccione la opción 6 para terminar el programa. Teclee ahora:

```
FOR I=0 TO 3:?LISTA$(I):NEXT[ENTER]
```

y debería ver:

```
AA1\AA2\
BB1\BB2\
CC1\CC2\
DD1\DD2\
```

Si todo ha ido bien, quizá desee arrancar el programa otra vez, ahora con un GOTO 10000, y utilizar la opción 5 del menú para almacenar en cinta los datos que ha introducido. Esto hará que las comprobaciones siguientes sean menos pesadas.

Módulo 4.1.10: Identificación de elementos en una entrada

Antes de que podamos pasar a los módulos que permiten la recuperación de los datos del fichero y su manipulación posterior, necesitamos introducir este otro, cuya tarea es buscar dentro de una entrada y registrar la posición de cada carácter «\» o, en realidad, el final de cada elemento de esa entrada. Los valores son almacenados en la matriz PTR, y solamente estarán relacionados con la entrada en vigor. Cuando se quiera otra entrada, este módulo tendrá que ser llamado de nuevo para analizarla.

Módulo 4.1.10: Líneas 19000 - 19080

```
19000 REM*******************************
19010 REM Analizar registro
19020 REM*******************************
19030 pp=0
19040 FOR i=0 TO x-1
19050 ptr(i)=INSTR(pp+1,lista$(s1),"\")
19060 pp=ptr(i)
19070 NEXT i
19080 RETURN
```

Comentarios
Línea 19050: Excepto en el módulo de búsqueda binaria, en el que la variable BB se utiliza para indicar la entrada que se está examinando, el resto del programa hace uso de la variable S1 para registrar la posición de la entrada actual en LISTA$. El efecto de esta línea es buscar cada presencia del carácter «\», comenzando un carácter después de la posición en que se encontró el último.

Módulo 4.1.11: Búsqueda de elementos en el fichero

Ahora podemos pasar al módulo que hace *útil* al programa, permitiendo la recuperación de los datos que han sido almacenados. Este módulo permitirá la recuperación de las entradas mediante uno de los siguientes cuatro métodos:

1) Uno a uno, según ordenación en efecto, desde la posición en vigor.
2) Saltándose hacia atrás o hacia adelante un número de elementos especificado.
3) Introduciendo un elemento clave (el primer elemento de la entrada) para conseguir una búsqueda rápida.

4) Buscando la presencia de una combinación de caracteres, dondequiera que esté dentro de una entrada.

Módulo 4.1.11: Líneas 17000 - 17430

```
17000 REM*******************************
17010 REM Busqueda
17020 REM*******************************
17030 s1=0:CLS:PRINT TAB(16);"BUSQUEDA":
PRINT TAB(16);"========"
17040 IF el=0 THEN PRINT "**AUN NO SE HA
 INTRODUCIDO NINGUN DATO**" ELSE GOTO 17
070
17050 FOR i=1 TO 2000:NEXT i
17060 RETURN
17070 PRINT "ORDENES DISPONIBLES:"
17080 PRINT:PRINT "- Introduzca un eleme
nto para busqueda    normal.":PRINT
17090 PRINT "- Anteponga '*' para busque
da inicial. ":PRINT
17100 PEN 2:PRINT "ENTER";
17110 PEN 1:PRINT " para ver el primer e
lemento del":PRINT "fichero.":PRINT
17120 te$="":PRINT:INPUT "Introduzca ord
en de busqueda: ",te$:te$=UPPER$(te$)
17130 IF LEFT$(te$,1)="*" THEN te$=MID$(
te$,2):GOSUB 13000:te$="":s1=bb
17140 p$="C":WHILE p$="C"
17150 IF te$="" THEN GOTO 17210
17160 ff=0:FOR i=s1 TO el-1
17170 pp=INSTR(lista$(i),te$)
17180 IF pp<>0 THEN ff=1:s1=i:i=el-1
17190 NEXT i
17200 IF ff=0 THEN RETURN
17210 p$=""
17220 WHILE p$="" AND el>0
17230 IF s1>el-1 THEN s1=el-1
17240 IF s1<0 THEN s1=0
17250 GOSUB 19000
17260 CLS:PRINT "ENTRADA ";s1+1;":":PRIN
T:pp=0
17270 FOR i=0 TO x-1
17280 PRINT elemento$(i);":";MID$(lista$
(s1),pp+1,ptr(i)-pp-1)
17290 pp=ptr(i):NEXT i
17300 PRINT:PRINT "ORDENES DISPONIBLES:"
17310 PRINT:PEN 2:PRINT "ENTER";:PEN 1:P
```

```
RINT " para ver el elemento siguiente.":
PRINT
17320 PRINT "'R' para rectificar.":PRINT
17330 PRINT "'C' para continuar busqueda
.":PRINT
17340 PRINT "'#' seguido de un numero pa
ra mover pun-     tero.":PRINT
17350 PRINT "'\' para abandonar busqueda
.":PRINT
17360 PRINT:INPUT "Cual elige: ",p$
17370 p$=UPPER$(p$):IF p$="" OR p$="C" T
HEN s1=s1+1
17380 IF p$="C" THEN GOTO 17420
17390 IF p$="R" THEN GOSUB 15000:p$=""
17400 IF LEFT$(p$,1)="#" THEN s1=s1+VAL(
MID$(p$,2)):p$=""
17410 WEND
17420 WEND
17430 RETURN
```

Comentarios

Línea 17030: S1, tal y como se mencionó en los comentarios del módulo anterior, es el puntero de la entrada actual, y se pone a 0 cada vez que se comienza una búsqueda.

Líneas 17040-17060: Se genera un mensaje de error en caso de que aún no se hayan introducido elementos en el fichero.

Líneas 17070-17120: Es el menú de activación del módulo de búsqueda. En cada búsqueda lo verá solamente una vez: al comienzo. Mediante este menú se puede especificar el tipo de búsqueda; si desea cambiar de tipo, tendrá que abandonar la búsqueda actual y comenzar de nuevo con este menú. El término «BUSQUEDA NORMAL» indica la búsqueda de una combinación dada de caracteres; la primera entrada proporcionada por este método será la primera del fichero que contenga esos caracteres, estén en la posición que estén. El término «BUSQUEDA INICIAL» se refiere a la búsqueda de una entrada que *comience con* la combinación de caracteres especificada por el usuario. Así, la introducción de «*LAN» haría que se encontrase una entrada que comenzara con las letras LAN, aunque no necesariamente la primera. Si la cadena especificada no está presente al comienzo de ninguna entrada, la entrada proporcionada será la que ocupa el lugar en que se insertaría si se tratase de una nueva entrada.

Una búsqueda normal o inicial se puede extender a más de un elemento de la entrada incluyendo un carácter «\» en la cadena a localizar. Utilizando esta técnica en un fichero en el que el primer elemento de cada entrada sea un apellido y el segundo un nombre, una

búsqueda inicial de «LANDABURU»\«J» encontraría cualquier LANDABURU con inicial «J» en el nombre, aunque no necesariamente el primero.

Si, durante una búsqueda normal, no se encuentra la cadena especificada en ninguna de las entradas del fichero, la ejecución del programa retornará al menú principal.

Línea 17130: Esta línea hace todo el trabajo necesario para una «búsqueda inicial», quitando el asterisco de la izquierda y llamando simplemente al módulo de «búsqueda binaria» para encontrar la posición correcta.

Líneas 17140-17420: El bucle principal que hará que se repita una búsqueda normal, si el usuario así lo solicita en un menú posterior. Observe que la rutina de búsqueda inicial no se encuentra dentro de este bucle, cosa lógica ya que no tiene sentido la repetición de ese tipo de búsqueda (el resultado sería siempre la misma entrada).

Líneas 17150-17210: En este punto de la ejecución del módulo, cualquier cosa introducida debe ser una cadena a localizar utilizando la búsqueda normal. Esto se hace de forma bastante sencilla explorando las entradas mediante INSTR. Si se encuentra un elemento, la variable FF se pone a 1 para indicar este hecho. La posición se registra en S1, y entonces la variable I del bucle se pone a su valor más alto con lo que el bucle finalizará.

Líneas 17220-17410: Este bucle continuará en tanto que P$, la entrada para el menú de búsqueda posterior, continúe siendo una «cadena nula». La razón por la que debe incluirse el valor de EL en la condición del bucle, es que dentro de este bucle habrá medidas para la supresión de elementos. Si se suprimen todos los elementos, necesitaremos salir del bucle o se generará un error.

Líneas 17230-17240: Dentro del bucle, el usuario tiene la opción de moverse por el fichero mediante números. Estas líneas comprueban, en las pasadas consecutivas por el bucle, que el puntero no se ha ido fuera del intervalo permitido por el número de entradas en vigor.

Líneas 17250-17290: Habiendo llamado al módulo anterior, estas líneas hacen uso de la información acerca de la posición de los caracteres separadores, para presentar los elementos que componen la entrada actual, junto con la denominación del elemento. Por cada pasada a través del bucle FOR, se presentan los caracteres con una posición intermedia entre el valor de la variable PP y un valor contenido en la matriz PTR. Una vez que se presenta cada lote de caracteres, PP es puesta al valor actual en la matriz PTR y la variable I del bucle coge el valor siguiente en PTR.

Líneas 17300-17360: El menú que aparece una vez se ha visualizado una entrada. El usuario tiene la opción de moverse a la entrada siguiente, de llamar a la función de «RECTIFICACION» (que aún no se

ha introducido), de continuar la búsqueda de la cadena especificada con anterioridad, de moverse por el fichero en saltos correspondientes a un número de entradas especificado o de volver al menú principal del programa.
Líneas 17370-17380: Estas dos líneas se relacionan con los bucles que comienzan en las líneas 17140 y 17220. Si se introduce una «cadena nula» (esto es, se pulsa ENTER), entonces se repite el bucle de la línea 17220, visualizándose la siguiente entrada indicada por S1. Si se introduce «C», el bucle de la línea 17140 ejecuta otra vez la búsqueda, comenzando por la entrada siguiente.
Línea 17390: La función de «RECTIFICACION», que aún no se ha introducido.
Línea 17400: La introducción de un número precedido de un símbolo «#», permite al usuario ir hacia adelante o hacia atrás por el fichero. El hacer P$ igual a una cadena nula hace simplemente que se ejecute el bucle de la línea 17220 para visualizar la entrada a la que se ha llegado.

Comprobación

Si ha grabado previamente la serie de cuatro entradas hecha en comprobaciones anteriores, lance el programa y cargue dichas entradas desde la cinta. Especifique la opción 4 en el menú principal y entonces, cuando aparezca en pantalla el menú de búsqueda, recorra las entradas pulsando ENTER. Cada entrada debería visualizarse en dos líneas, con la denominación de elemento apropiada, por ejemplo:

```
UNO: AA1
DOS: AA2
```

Cuando llegue a la entrada cuarta encontrará que no se puede seguir adelante mediante más pulsaciones de la tecla ENTER. Teclee ahora «#–1», con lo que debería retroceder a la entrada 3. Continúe retrocediendo, encontrará que tampoco puede retroceder más allá del comienzo del fichero.

Pulse «\» para volver al menú principal y especifique otra vez la opción 4. Esta vez responda con «CC» al menú de búsqueda inicial: se debería visualizar la entrada 3 (la única que contiene los caracteres «CC»). Ahora está en el segundo menú de búsqueda, así pues pulse «C» para continuar la búsqueda. En respuesta, debería encontrarse otra vez frente al menú principal.

Escoja la opción 4 una vez más, pero esta vez teclee «2» como objeto de búsqueda. Debería aparecer la entrada 1, ya que contiene el carácter «2». Pulse «C» para continuar la búsqueda, con lo que se debería visualizar la entrada 2 (también contiene el carácter 2). Con-

tinúe pulsando «C» hasta que se hayan visualizado las cuatro entradas y fracase la búsqueda devolviéndole al menú principal.

Por último, escoja la opción 4 en el menú principal y pulse «*B» como su objeto de búsqueda. Debería visualizarse la entrada 2 (la única entrada que comienza por «B»). Pulse «\» para volver al menú principal y finalice el programa.

Acaba de comprobar todas las funciones de búsqueda.

Módulo 4.1.12: Anulación de entradas

Vamos a dar los últimos toques al programa añadiendo los dos módulos que vienen a continuación, módulos que permiten cambiar o anular las entradas. Primero se añade el módulo de «ANULACION», ya que será utilizado siempre que se cambie una entrada.

Módulo 4.1.12: Líneas 16000 - 16040

```
16000 REM*******************************
16010 REM Anular elemento
16020 REM*******************************
16030 FOR j=s1 TO el-1:lista$(j)=lista$(
j+1):NEXT j
16040 el=el-1:RETURN
```

Comentarios
Líneas 16030-16040: La anulación se efectúa simplemente comenzando por la entrada que hay por encima de la que se va a anular, y copiándola un lugar más abajo. Cuando cada entrada se ha movido un lugar hacia abajo, se reduce en 1 la cuenta de elementos.

Comprobación

Lance el programa y cargue los cuatro elementos desde la cinta. Especifique la opción 6 para terminar el programa. Ahora teclee:

S1=0[ENTER]
GOTO 16000

El programa debería detenerse con un mensaje de error «Unexpected RETURN». Teclee:

GOTO 11000

y entonces llame a la opción de búsqueda. Debería encontrarse con que la entrada AA1\AA2\ ha desaparecido del fichero.

Módulo 4.1.13: Cambio de entradas

El programa tendría un uso reducido si no pudiéramos cambiar los datos existentes. Este módulo llena ese hueco.

Módulo 4.1.13: Líneas 15000 - 15190

```
15000 REM*******************************
15010 REM Cambiar entrada
15020 REM*******************************
15030 te$="":pp=0
15040 FOR i=0 TO x-1
15050 CLS:PRINT "Entrada ";si+1;":"
15060 PRINT:PRINT elemento$(i);":";MID$(
lista$(si),pp+1,ptr(i)-pp-1)
15070 PRINT:PRINT "ORDENES DISPONIBLES:"
:PRINT
15080 PEN 2:PRINT "ENTER";
15090 PEN 1:PRINT " deja el elemento ina
lterado.":PRINT
15100 PRINT "Introduzca el nuevo element
o que reemplace al que se muestra.":PRIN
T
15110 PRINT "'\S' suprime toda la entrad
a.":PRINT
15120 PRINT "'\' deja toda la entrada in
alterada.":PRINT
15130 PRINT:INPUT "Cual desea: ",q$
15140 q$=UPPER$(q$):IF q$="\S" THEN GOSU
B 16000:RETURN
15150 IF q$="\" THEN RETURN
15160 IF q$<>"" THEN q$=q$+"\"
15170 IF q$="" THEN q$=MID$(lista$(si),p
p+1,ptr(i)-pp)
15180 pp=ptr(i):te$=te$+q$:NEXT i:GOSUB
16000:GOSUB 13000
15190 si=bb:GOSUB 14000:RETURN
```

Comentarios
Líneas 15040 - 15180: Este bucle, aunque parece muy similar al bucle que presenta las entradas en el módulo 4.1.8, es diferente en que únicamente presenta un elemento cada vez.
Línea 15140: La pulsación de «\S», cuando se está visualizando un elemento, suprime la totalidad de la entrada a la que pertenece ese elemento (observe que no se puede suprimir un *elemento* individual ya que el número de elementos por entrada es fijo).

Línea 15150: La pulsación de «\» en respuesta a cualquier elemento devuelve la ejecución al módulo de búsqueda. Se ignorarán cualesquiera cambios hechos sobre los elementos previos de la misma entrada, por lo que la entrada permanecerá inalterada.
Línea 15160: Si se hace una entrada distinta de «\S» o de «\» se interpreta como el sustituto del elemento que se está visualizando. Se añade el separador «\» al final del elemento, igual que se hacía en el módulo de nuevas entradas.
Línea 15170: La pulsación de ENTER, sin ninguna entrada, copia sin cambios al elemento visualizado. Si solamente se va a cambiar un elemento, simplemente pulse ENTER para los demás. Observe la única diferencia entre la expresión de cadena que hay aquí, y la utilizada en la línea 15060 para presentar el elemento. Falta el «–1» del final, con lo que no se quita el carácter separador «\».
Líneas 15180-15190: La entrada rectificada es montada en TE$. Cuando la entrada está completa se suprime la entrada original del fichero. La razón para hacerlo así, es que los cambios hechos pueden haber alterado la posición correcta de la entrada en el fichero. Una vez suprimida la entrada es enviada al módulo de búsqueda binaria y reinsertada, habiéndose copiado su posición dentro del fichero en la variable S1, por lo que el módulo de búsqueda sabrá qué elemento visualizar si se ha cambiado la posición.

Comprobación

Lance el programa y cargue los cuatro elementos de datos desde la cinta. Llame a la opción de búsqueda y pulse ENTER para que se visualice la entrada 1. Ahora pulse «R» en respuesta al segundo menú de búsqueda, con lo que debería ver visualizado el primer elemento («AA1») junto con el menú de RECTIFICACION, teclee «AAA1» y, cuando se visualice el segundo elemento, pulse ENTER. Debería haber vuelto al menú de búsqueda, y la entrada debería estar visualizada como:

```
UNO: AAA1
DOS: AA2
```

Intente hacer otros cambios y anulaciones en las entradas.

PROGRAMA 4.2: Nnúmero

Función del programa

No todas las clasificaciones están relacionadas con palabras. Una de las cosas que mejor hacen los microordenadores es almacenar y manipular números. El programa del que nos vamos a ocupar ahora, «NNUMERO» (una abreviación de Nombre y Número), le permite almacenar los nombres de ciertos elementos, las unidades en que se suelen medir y una cantidad asociada a cada uno de ellos. Ahora, antes de que diga que no ve ninguna utilidad a tal programa, considere al comerciante medio o incluso a quien hace la comida en casa.

El comerciante tiene un montón de artículos a los que se les conoce como existencias o stock. Todos los artículos que componen el stock tienen nombres, se presentan según diferentes unidades (caja, botella, saco, etc.), y todos tienen una cantidad muy importante asociada a ellos: su precio. Por consiguiente, para ayudarse de un microordenador en la elaboración de inventarios o en la redacción de una factura, el micro debe poder recordar estos tres datos sobre cada uno de los artículos. En casa, cada uno de los alimentos que comemos tiene un nombre, se presentan en diferentes unidades (cucharadas, kilos, pellizcos, etc.) y, si estamos interesados en sus efectos sobre nuestro peso, todos tienen una cantidad asociada conocida como «calorías».

Esto son solamente dos ejemplos (usted mismo puede imaginar muchos más) de la importancia de poder registrar nombres, unidades y una cantidad asociada en un surtido completo de artículos.

El objetivo de Nnúmero es permitirle crear un «diccionario» de artículos (hasta 1000) junto con las unidades en que se miden y los valores asociados con esas unidades. Basándose en ese diccionario podrá construir listas de artículos, que el programa presentará y totalizará por usted. Nnúmero resulta tan sencillo de utilizar para calcular la suma de las calorías de la receta de un día, como para suministrar el precio total de un conjunto de artículos.

Módulo 4.2.1: Inicialización

Un módulo estándar, únicamente merece la pena observar que se le pide al usuario que especifique el tipo de artículo con el que tratará el programa. El tipo será un nombre tal como «alimento» o «artículo de stock». La frase introducida se utilizará a lo largo del programa para solicitar del usuario que introduzca otro artículo.

```
10000 REM*******************************
10010 REM Inicializacion
10020 REM*******************************
10030 MODE 1
10040 DIM lista$(1000,1),lista(1000),te$
(100,1),te(100):ac=0:ar=0
10050 PEN 2:PRINT TAB(17);"NNUMERO":PRIN
T TAB(17);"========"
10060 WINDOW 1,40,4,25
10070 PEN 1
10080 INPUT "Desea cargar desde la cinta
 (s/n)";q$
10090 IF LOWER$(q$)="s" THEN GOSUB 22000
 ELSE PRINT:INPUT "Denominacion global d
e los articulos:    ",nn$
```

```
                NNUMERO
                =======

    ARTICULO:CORCHETES
    Unidades:3 CAJA
    Cantidad: 1656
    ---------------------------------------
    ARTICULO:RIBETES
    Unidades:11 CAJA
    Cantidad: 43120
    ---------------------------------------
    ARTICULO:BOTONES
    Unidades:6 BOLSA
    Cantidad: 16440
    ---------------------------------------
    ARTICULO:CREMALLERAS
    Unidades:35 PAQUETE
    Cantidad: 42400
    ---------------------------------------

    Total: 103616

    Pulse cualquier tecla para ir al menu
```

Fig. 4.2 Nnúmero en modo de presentación de la lista actual.

Comentarios

Línea 10040: La matriz LISTA$ se utiliza para registrar el nombre del artículo y de la unidad de cada artículo del diccionario. La cantidad

asociada a cada unidad se almacena en el elemento equivalente de la matriz LISTA.

TE$ y TE desempeñarán un objetivo similar al de LISTA$ y LISTA, pero para la «lista actual» que es extraída del diccionario. La variable AC registrará el número de artículos de la «lista actual» (la lista procedente del diccionario principal). AR, como en la mayoría de los programas del libro, registra el número de artículos del fichero principal (en este caso: el diccionario).

Módulo 4.2.2: Menú

Un módulo estándar de menú, cuya única diferencia respecto a los de los programas anteriores es la forma en que la línea 11160 impide que se pueda acceder a ciertas funciones del programa, antes de que se haya introducido algún dato (aunque tal posibilidad no se podrá utilizar hasta que no se introduzca el módulo siguiente).

Módulo 4.2.2: Líneas 11000 - 11210

```
11000 REM*******************************
11010 REM Menu principal
11020 REM*******************************
11030 WHILE z<>8
11040 CLS
11050 PRINT "ORDENES DISPONIBLES:"
11060 PRINT
11070 PRINT "1) Presentar la lista actua
l."
11080 PRINT "2) Introducir en la lista a
ctual."
11090 PRINT "3) Empezar una lista nueva.
"
11100 PRINT "4) Borrar de la lista actua
l."
11110 PRINT "5) Ampliar diccionario."
11120 PRINT "6) Examinar articulos del d
iccionario."
11130 PRINT "7) Grabar datos en cinta."
11140 PRINT "8) Terminar."
11150 PRINT:INPUT "Cual elige:",z
11160 IF (z=1 OR z=2 OR z=4 OR z=6 OR z=
7) AND ar=0 THEN x$="****AUN NO SE HAN I
NTRODUCIDO DATOS*****":GOSUB 23000:z=0
11170 ON z GOSUB 12000,13000,14000,20000
,15000,18000,21000
```

```
11180 WEND
11190 CLS
11200 LOCATE 11,11:PRINT "PROGRAMA TERMI
NADO"
11210 END
```

Módulo 4.2.3: Mensajes de error

En caso de que un programa sea capaz de generar un cierto número de mensajes de error para el usuario, el pequeño módulo que viene a continuación puede suministrar un ahorro considerable de memoria. Todo lo que hace el módulo es presentar una cadena denominada X$, emitir un pitido de corta duración y retornar entonces al lugar desde donde se le llamó. Si examina la línea 11160 verá que, para llamar a este módulo, lo que se hace es definir X$ y entonces dar una orden GOSUB

Módulo 4.2.3: Líneas 23000-23070

```
23000 REM*******************************
23010 REM Error
23020 REM*******************************
23030 PRINT:PRINT x$
23040 SOUND 1,1000,100
23050 FOR i=1 TO 1500
23060 NEXT i
23070 RETURN
```

Comprobación

Lance el programa y responda de forma adecuada a las distintas peticiones. Cuando llegue al menú especifique la opción 1, y se presentará inmediatamente el mensaje de error, indicando que aún no hay datos en la memoria.

Módulos 4.2.4 y 4.2.5: Ficheros de datos

Dos módulos estándar.

Módulo 4.2.4 y 4.2.5: Líneas 21000 - 22210

```
21000 REM*******************************
21010 REM Grabar en cinta
21020 REM*******************************
21030 CLS
21040 PRINT "Grabar datos:":PRINT
```

```
21050 INPUT "Nombre del fichero:",ficher
o$
21060 OPENOUT fichero$
21070 PRINT #9,nn$
21080 PRINT #9,ac
21090 PRINT #9,ar
21100 FOR i=0 TO ac-1
21110 PRINT #9,te$(i,0)
21120 PRINT #9,te$(i,1)
21130 PRINT #9,te(i)
21140 NEXT i
21150 FOR i=0 TO ar-1
21160 PRINT #9,lista$(i,0)
21170 PRINT #9,lista$(i,1)
21180 PRINT #9,lista(i)
21190 NEXT i
21200 CLOSEOUT
21210 RETURN
22000 REM*******************************
22010 REM Cargar desde cinta
22020 REM*******************************
22030 CLS
22040 PRINT "Cargar datos:":PRINT
22050 INPUT "Nombre del fichero:",ficher
o$
22060 OPENIN fichero$
22070 INPUT #9,nn$
22080 INPUT #9,ac
22090 INPUT #9,ar
22100 FOR i=0 TO ac-1
22110 INPUT #9,te$(i,0)
22120 INPUT #9,te$(i,1)
22130 INPUT #9,te(i)
22140 NEXT i
22150 FOR i=0 TO ar-1
22160 INPUT #9,lista$(i,0)
22170 INPUT #9,lista$(i,1)
22180 INPUT #9,lista(i)
22190 NEXT i
22200 CLOSEIN
22210 RETURN
```

Módulo 4.2.6: Búsqueda binaria

Si desea unos comentarios detallados vea el módulo equivalente de Uniarchivo. La única cosa a tener en cuenta en el módulo es que

la clasificación se hace en base al nombre del artículo de la columna cero de la matriz LISTA$.

Módulo 4.2.6: Líneas 16000 - 16110

```
16000 REM*******************************
16010 REM Busqueda binaria
16020 REM*******************************
16030 IF ar=0 THEN bb=0:RETURN
16040 po=INT(LOG(ar)/LOG(2)):bb=2^po-1
16050 FOR i=po TO 0 STEP -1
16060 bb=bb+2^i*((lista$(bb,0)>t1$)-(lis
ta$(bb,0)<t1$))
16070 IF bb<0 THEN bb=0
16080 IF bb>ar-1 THEN bb=ar-1
16090 NEXT i
16100 IF lista$(bb,0)<t1$ THEN bb=bb+1
16110 RETURN
```

Módulo 4.2.7: Inserción de artículos en el diccionario principal

El fundamento de este módulo es exactamente el mismo que el del módulo análogo de Uniarchivo. La razón por la que este módulo es ligeramente más largo es que tiene que insertar dos cadenas y un número en las dos matrices, en vez de una única cadena.

Módulo 4.2.7: Líneas 17000 - 17110

```
17000 REM*******************************
17010 REM Insertar articulo
17020 REM*******************************
17030 FOR i=ar TO bb+1 STEP -1
17040 lista$(i,0)=lista$(i-1,0)
17050 lista$(i,1)=lista$(i-1,1)
17060 lista(i)=lista(i-1)
17070 NEXT i
17080 lista$(bb,0)=t1$
17090 lista$(bb,1)=t2$
17100 lista(bb)=nn
17110 RETURN
```

Módulo 4.2.8: Introducción de artículos para el diccionario

Considerablemente menos complicado que el módulo equivalente de Uniarchivo, este módulo acepta tres entradas del usuario: (a)

el nombre del artículo, (b) el nombre de las unidades en que se mide y (c) la cantidad asociada a esas unidades.

Módulo 4.2.8: Líneas 15000 - 15170

```
15000 REM*******************************
15010 REM Ampliar diccionario
15020 REM*******************************
15030 WHILE 1
15040 CLS
15050 PRINT "Nuevos articulos para el di
ccionario:    ":PRINT
15060 IF ar>1000 THEN x$="***** NO SE AD
MITEN MAS ARTICULOS ******":GOSUB 23000:
RETURN
15070 q$="":WHILE LOWER$(q$)<>"s"
15080 PRINT nn$;:INPUT " (introducir nom
bre o '\' para volver al menu):",t1$
15090 IF t1$="\" THEN RETURN
15100 PRINT "unidades";:INPUT ":",t2$
15110 PRINT "Cantidad por ";LOWER$(t2$);
:INPUT ":",nn
15120 PRINT:INPUT "Es correcto (s/n)";q$
:PRINT
15130 WEND
15140 GOSUB 16000
15150 GOSUB 17000
15160 ar=ar+1
15170 WEND
```

Comprobación

Por fin se hace posible hacer una comprobación de todo lo que se ha introducido hasta ahora.

Lance el programa, especifique que no va a cargar desde la cinta, y dé el nombre ARTICULO en respuesta a la petición de denominación global. Una vez en el menú principal, escoja la opción 5 (Ampliar diccionario). Cuando aparezca la pantalla de «nuevos artículos», meta las tres entradas siguientes:

COSA1/CAJAS/10
COSA2/BOTELLAS/20
COSA3/SACOS/40

Estos artículos no tienen ningún significado en particular, son simplemente para efectuar la comprobación.

Cuando haya introducido los artículos vuelva al menú principal pulsando «\». Ahora llame al módulo de fichero de datos (opción 7) para almacenar la información. Termine el programa con la opción 8 del menú, e introduzca:

```
FOR I=0 TO 2:?LISTA$(I,0),LISTA$(I,1),LISTA(I):NEXT[ENTER]
```

Debería ver esto:

```
COSA1       CAJAS       10
COSA2       BOTELLAS    20
COSA3       SACOS       40
```

Ahora lance el programa y especifique qué *desea* cargar desde la cinta. Dé el nombre que suministró al almacenar los datos. Cuando haya terminado la carga desde la cinta debería poder realizar la misma prueba con los contenidos de las matrices, obteniendo el mismo resultado.

Módulo 4.2.9: La rutina de búsqueda

Como en Uniarchivo, este módulo proporciona al usuario la oportunidad de moverse por el fichero de artículos del diccionario, para buscar artículos con determinado nombre o eliminarlos del fichero. El módulo es más sencillo que el que se dió en Uniarchivo, ya que está diseñado para buscar solamente artículos completos, en vez de combinaciones de caracteres colocados en algún lugar de un artículo. Además, la estructura de una entrada completa en Nnúmero, es bastante más simple que la estructura de un elemento de la matriz principal de Uniarchivo.

Módulo 4.2.9: Líneas 18000 - 18250

```
18000 REM*****************************
18010 REM Busqueda por parte del usuario
18020 REM*****************************
18030 bb=0
18040 t1$="":WHILE ar>0
18050 CLS:PRINT "Busqueda:":PRINT
18060 PRINT "Numero del articulo:";bb+1
18070 PRINT nn$;":";lista$(bb,0)
18080 PRINT "Unidades:";lista$(bb,1)
```

```
18090 PRINT "Cantidad en ";LOWER$(lista$
(bb,1));":";lista(bb)
18100 PRINT:PRINT "Ordenes disponibles:"
:PRINT
18110 PRINT "Introduzca el articulo a bu
scar"
18120 PEN 2:PRINT "ENTER";:PEN 1:PRINT "
 para ver el articulo siguiente"
18130 PRINT "'#' y un numero para mover
el puntero"
18140 PRINT "'b' para borrar el articulo
"
18150 PRINT "'\' para volver al menu"
18160 PRINT:INPUT "Cual desea";t1$
18170 IF t1$="" THEN t1$="#1"
18180 IF t1$="\" THEN RETURN
18190 IF LOWER$(t1$)="b" THEN GOSUB 1900
0:t1$=""
18200 IF LEFT$(t1$,1)="#" THEN bb=bb+VAL
(MID$(t1$,2)):t1$=""
18210 IF t1$<>"" THEN GOSUB 16000
18220 IF bb>ar-1 THEN bb=ar-1
18230 IF bb<0 THEN bb=0
18240 WEND
18250 RETURN
```

Comentarios

Línea 18170: En vez de tomar medidas especiales para la introducción de un ENTER (es decir: una cadena vacía), lo que se hace es poner primero T1$ a #1. Si el usuario pulsa simplemente ENTER, el contenido de T1$ permanecerá inalterado y se mostrará el siguiente artículo del fichero.

Comprobación

Simplemente lance el programa, cargue los tres artículos desde la cinta, y entonces llame a la opción 6 (Mostrar diccionario) del menú principal. Debería poder recorrer los artículos, hacia adelante o hacia atrás, utilizando un número precedido de # como en Uniarchivo. También debería poder recuperar un artículo introduciendo su nombre (no el nombre de la unidad).

Módulo 4.2.10: Borrado de un artículo

El equivalente directo del módulo correspondiente de Uniarchivo.

Módulo 4.2.10: Líneas 19000 - 19090

```
19000 REM*******************************
19010 REM Borrar
19020 REM*******************************
19030 FOR i=bb TO ar-2
19040 lista$(i,0)=lista$(i+1,0)
19050 lista$(i,1)=lista$(i+1,1)
19060 lista(i)=lista(i+1)
19070 NEXT i
19080 ar=ar-1
19090 RETURN
```

Comprobación

Lance el programa, cargue los datos, llame a la opción 6 desde el menú principal y pulse «B» frente a una de las entradas. Debería encontrar que la entrada ha sido suprimida del fichero.

Módulo 4.2.11: Copia de artículos en la lista actual

El objetivo de Nnúmero no es simplemente el mantenimiento de un diccionario de artículos, y de sus cantidades asociadas, sino la utilización de ese diccionario como la base sobre la que poder construir listas provisionales. Por consiguiente, los módulos que vienen a continuación están diseñados para permitir que el usuario pueda añadir artículos a la lista «actual», visualizar esa lista, suprimir artículos individuales o borrar toda la lista de una única operación. Este módulo permite la copia de artículos desde el diccionario a la lista «actual».

Módulo 4.2.11: Líneas 13000 - 13210

```
13000 REM*******************************
13010 REM Ampliar lista actual
13020 REM*******************************
13030 WHILE 1
13040 CLS
13050 PRINT "Ampliaciones a la lista act
ual:":PRINT
13060 IF ac>100 THEN x$="**** LA LISTA A
CTUAL ESTA COMPLETA *****":GOSUB 22000:R
ETURN
13070 PRINT nn$;:INPUT " ('\' para volve
r al menu):",t1$
13080 IF t1$="\" THEN RETURN
13090 conocido=0:GOSUB 16000:IF lista$(b
b,0)=t1$ THEN conocido=1
```

```
13100 IF conocido=0 THEN x$="**** "+UPPE
R$(nn$)+" DESCONOCIDO, COMPRUEBE POR FAV
OR ****":GOSUB 23000:RETURN
13110 PRINT:PRINT "Unidades:";lista$(bb,
1)
13120 INPUT "Cantidad:",q
13130 PRINT:INPUT "Es correcto (s/n)";q$
13140 WHILE LOWER$(q$)="s"
13150 te$(ac,0)=lista$(bb,0)
13160 te$(ac,1)=MID$(STR$(q),2)+" "+list
a$(bb,1)
13170 te(ac)=q*lista(bb)
13180 ac=ac+1
13190 q$=""
13200 WEND
13210 WEND
```

Comentarios

Líneas 13090-13100: Estas líneas realizan una comprobación para ver si el artículo introducido por el usuario (el que se va a colocar en la lista actual) está presente en el diccionario principal. Esto se hace llamando al módulo de búsqueda binaria, para obtener la posición en la que el artículo sería insertado en el diccionario. El contenido real del diccionario en este punto, es entonces comparado con lo que el usuario ha introducido. Si el artículo introducido por el usuario está en el diccionario, entonces los dos artículos serán iguales, en caso contrario se presenta un mensaje de error y el módulo termina.

Línea 13110: Habiendo encontrado el artículo en el diccionario, el módulo presenta las unidades en que se suele medir, y pregunta cuántas de esas unidades hay que incluir.

Líneas 13120-13170: Se introducen las cantidades concretas. Entonces se le pide al usuario que confirme la exactitud de la entrada, antes de añadirla a la lista actual (contenida en las variables TE$ y TE). Observe que la cantidad almacenada en TE, no es la cantidad por unidad tomada del diccionario, sino la cantidad *total* correspondiente al número de unidades especificado por el usuario.

Comprobación

Lance el programa y cargue los datos desde la cinta. Llame a la opción 2 del menú principal. Introduzca los artículos y el número de unidades que se exponen a continuación:

COSA1,1
COSA2,2
COSA3,3

Ahora trate de conseguir que el módulo acepte «COSA4» (que no está presente en el diccionario). Debería recibir un mensaje de error, pidiéndole que compruebe el nombre del artículo. Antes de hacer cualquier otra cosa llame a la opción 7 para grabar la lista actual que acaba de crear, junto con el diccionario principal. Finalmente, escoja la opción 8 para terminar el programa.

Ahora introduzca la línea siguiente en modo directo:

```
FOR I=0 TO 2:?TE$(I,0),TE$(I,1),TE(I):NEXT[ENTER]
```

con lo que debería ver lo siguiente:

```
COSA1        1 CAJAS        10
COSA2        2 BOTELLAS     40
COSA3        3 SACOS        120
```

Módulo 4.2.12: Presentación de la lista actual

El único propósito de este módulo es visualizar, una a una sobre la pantalla, las entradas que componen la lista actual. Tras cada entrada se le pide al usuario que pulse una tecla antes de visualizar la siguiente. Esto se hace así porque la lista será normalmente más larga que la profundidad de la pantalla, y el usuario no deseará que la lista se desplace verticalmente (scrolling) más rápidamente de lo que puede leer. Al final de la lista se da el total de las cantidades asociadas a los elementos componentes de la lista actual.

Módulo 4.2.12: Líneas 12000 - 12160

```
12000 REM*******************************
12010 REM Presentar lista actual
12020 REM*******************************
12030 IF ac=0 THEN RETURN
12040 CLS:ct=0
12050 FOR i=0 TO ac-1
12060 PRINT nn$;":";te$(i,0)
12070 PRINT "Unidades:";te$(i,1)
12080 PRINT "Cantidad:";te(i)
12090 PRINT "--------------------------
------------"
12100 WHILE INKEY$="":WEND
12110 ct=ct+te(i)
12120 NEXT i
12130 PRINT:PRINT "Total:";ct
```

```
12140 PRINT:PRINT "Pulse cualquier tecla
 para ir al menu"
12150 WHILE INKEY$="":WEND
12160 RETURN
```

Módulo 4.2.13: Borrado de artículos de la lista actual

Se trata de una versión mucho más simplificada del tipo de módulo de búsqueda utilizado para el diccionario principal. Permite al usuario avanzar por la lista actual, artículo por artículo, y borrar cualquier artículo pulsando «B» frente a él. El proceso se puede terminar en cualquier momento pulsando «\».

Módulo 4.2.13: Líneas 20000 - 20250

```
20000 REM*******************************
20010 REM Borrados en la lista actual
20020 REM*******************************
20030 i=0
20040 WHILE i<ac
20050 CLS
20060 PRINT "Borrados en lista actual:":
PRINT
20070 PRINT te$(i,0)
20080 PRINT te$(i,1)
20090 PRINT:PRINT "Ordenes disponibles:"
:PRINT
20100 PEN 2:PRINT "ENTER";:PEN 1:PRINT "
 para ver articulo siguiente"
20110 PRINT "'b' para borrar"
20120 PRINT "'\' para volver al menu"
20130 PRINT:INPUT "Cual desea";q$
20140 IF q$="\" THEN RETURN
20150 IF LOWER$(q$)<>"b" THEN i=i+1
20160 WHILE LOWER$(q$)="b"
20170 FOR j=1 TO ac-1
20180 te$(j,0)=te$(j+1,0)
20190 te$(j,1)=te$(j+1,1)
20200 te(j)=te(j+1)
20210 NEXT j
20220 ac=ac-1
20230 q$="":WEND
20240 WEND
20250 RETURN
```

Comentarios

Líneas 20150-20230: «I» es la variable que controla cuál de los artículos es visualizado. Observe que *no* se incrementa si se pulsa «B». «I» estará apuntando al artículo a borrar, y el proceso de borrado copiará (trasladará un lugar hacia abajo) el siguiente artículo en esa posición.

Comprobación

Lance el programa y cargue el fichero de datos que contiene a la lista actual (fichero que grabó en la comprobación anterior). Llame a la opción 4 (Borrar de la lista actual) desde el menú principal. Ahora debería poder recorrer los tres artículos de la lista actual y borrar uno. Se puede comprobar que el dato ha sido efectivamente borrado sin más que mostrar la lista actual.

Módulo 4.2.14: Inicialización de la lista actual

En muchas aplicaciones lo que se necesitará será construir una lista, obtener el total implicado, y entonces pasar rápidamente a una lista nueva. Este módulo sencillo borra el contenido de la lista actual en una sola operación:

Módulo 4.2.14: Líneas 14000 - 14090

```
14000 REM*******************************
14010 REM Inicializar lista actual
14020 REM*******************************
14030 FOR i=0 TO ac-1
14040 te$(i,0)=""
14050 te$(i,1)=""
14060 te(i)=0
14070 NEXT i
14080 ac=0
14090 RETURN
```

Comprobación

Lance el programa, cargue el fichero de datos que incluye la lista actual, y especifique la opción 3 (Empezar una lista nueva) desde el menú principal. Dicho menú debería volver a presentarse casi instantáneamente. Ahora intente llamar a la opción 1 desde el menú principal. Una vez más, todo lo que ocurre es que el menú vuelve a presentarse (se ha llamado al módulo de presentación pero, como no hay nada que mostrar, la ejecución retorna inmediatamene).

Si esta comprobación es totalmente satisfactoria, el programa está completo y listo para ser usado.

PROGRAMA 4.3: Editexto

Función del programa

Una de las aplicaciones más fascinantes del ordenador moderno es el procesamiento de palabras (tratamiento de textos). El uso de un procesador de textos moderno y sofisticado elimina el trabajo pesado que siempre ha acompañado a cualquier tipo de escritura, e incluso los documentos complejos se convierten en una tarea relativamente simple. No obstante, y desgraciadamente, un programa que corra un procesador de palabras con especificaciones plenas puede ocupar más memoria de la que se puede disponer en micros domésticos aún más potentes que el 464. Todavía más importante, rápido como es el BASIC del 464, la velocidad a que debe ejecutarse normalmente una aplicación como el tratamiento de palabras, requiere la programación directa en código máquina, el oscuro lenguaje entendido por el chip procesador Z80 del 464.

El programa que se da a continuación, Editexto, no puede ser alabado con el nombre de «procesador de palabras», aunque le permite tratar a su 464 como una forma más flexible de máquina de escribir: componiendo textos, cambiándolos, suprimiendo líneas o palabras, recorriendo líneas e imprimiéndolas en una impresora compatible. Si está buscando algo que se ocupe de una oficina no es esto lo que necesita, pero si lo que desea es algo para componer pequeños

```
                                        Una calle 123
                                        Una ciudad
                                        Tfno. 01 234 5678

Estimado Sr.:

  Gracias por su carta referente al desarrollo de Editexto como al-
ternativa comercial al Wordstar, pero creo que aún le falta un buen
recho por recorrer.

  Como puede ver, los recursos son bastante limitados. No obstante
esuelve la papeleta.

  Quedo en espera de sus noticias.

  Sinceramente suyo.

                                        David Lawrence
```

Fig. 4.3 Carta de muestra compuesta e impresa mediante Editexto.

documentos, entonces Editexto es una herramienta útil que le hará el trabajo; realmente, uno de mis mayores placeres como escritor es cuando recibo cartas de lectores de libros anteriores, mencionando que la carta ha sido compuesta e impresa utilizando versiones previas de Editexto.

Entre los nuevos conceptos introducidos en este programa se incluyen:

1) Cursores controlados por programa e introducción de texto.
2) Manipulación de datos en grandes matrices de cadena.
3) Salida a impresora de un texto complejo.

Módulo 4.3.1: Inicialización

Un módulo sencillo, con nada especial excepto el tratamiento de una o dos de las variables declaradas.

Módulo 4.3.1: Líneas 11000 - 11100

```
11000 REM*******************************
11010 REM Inicializacion
11020 REM*******************************
11030 in=1
11040 DIM texto$(100):ul=1:lu=1
11050 texto$(0)=STRING$(32,CHR$(245))
11060 texto$(1)=STRING$(32,CHR$(244))
11070 a$=" "
11080 INPUT "Desea cargar desde la cinta
 (s/n)";q$
11090 IF LOWER$(q$)="s" THEN GOSUB 19000
11100 RETURN
```

Comentarios
Línea 11040: La matriz TEXTO$ se utilizará para guardar la parte principal del texto que se haya introducido.
Líneas 11050-11060: Estas dos líneas de símbolos gráficos constituyen dos marcadores que aparecerán en pantalla cada vez que se tropiece con la parte superior o la inferior del bloque de texto.

Módulo 4.3.2: Introducción de caracteres

La primera tarea principal del programa será aceptar una entrada desde el teclado y *hacer* algo con ella, y esa es la tarea que realizará

este módulo. La mayoría de las líneas del módulo están relacionadas con la introducción de caracteres de «orden» especiales, caracteres que le dicen a Editexto que realice una tarea específica, como borrar un carácter. Observará, también, que hay algunas líneas que le serán familiares por el capítulo de gráficos, cuando tuvo que suministrarse un cursor móvil.

La función global del módulo es permitirle que introduzca texto en la línea 23 de la pantalla, editando conforme vaya avanzando. Módulos posteriores cogerán estas líneas y las incorporarán a la parte principal del texto.

Debería observar, también, que éste es uno de los pocos programas del libro que tiene separaciones entre las líneas para conseguir una mayor claridad. La razón es que, aunque corto, el programa utiliza una cantidad de nombres de variables y bucles pequeños, lo que hace que resulte muy difícil seguir el flujo sin una guía visual. Como en los programas anteriores que se han presentado de esta forma, las líneas de programa que contienen solamente el símbolo de dos puntos (:) pueden omitirse si lo desea.

Módulo 4.3.2: Líneas 12000 - 12430

```
12000 REM*******************************
12010 REM Editar linea
12020 REM*******************************
12030 :
12040 :
12050 WHILE 1
12060 :
12070 :
12080 t$="":WHILE t$=""
12090 LOCATE p-40*INT(p/40)+1,23+INT(p/4
0):PEN 1:PRINT CHR$(143)
12100 FOR tt=1 TO 20:NEXT tt
12110 LOCATE p-40*INT(p/40)+1,23+INT(p/4
0):PEN 2:PRINT MID$(a$,p+1,1)
12120 t$=INKEY$
12130 WEND
12140 :
12150 :
12160 IF t$=CHR$(13) OR (LEN(a$)>40 AND
t$=" ") THEN t$="":GOSUB 13000
12170 IF t$="^" THEN GOSUB 15000
12180 :
12190 :
12200 WHILE t$="\"
12210 IF p<>0 THEN t=0 ELSE t=LEN(a$)-1
```

```
12220 p=t
12230 t$="":WEND
12240 :
12250 :
12260 WHILE p>0 AND t$=CHR$(127)
12270 a$=LEFT$(a$,p-1)+MID$(a$,p+1)
12280 p=p-1
12290 t$="":WEND
12300 :
12310 :
12320 WHILE t$>=CHR$(32) AND t$<=CHR$(16
3) AND t$<>CHR$(127)
12330 a$=LEFT$(a$,p)+t$+MID$(a$,p+1)
12340 p=p+1
12350 t$="":WEND
12360 :
12370 :
12380 IF t$=CHR$(242) AND p>0 THEN p=p-1
12390 IF t$=CHR$(243) AND p<LEN(a$)-1 TH
EN p=p+1
12400 :
12410 :
12420 LOCATE 1,23:PRINT a$
12430 WEND
```

Comentarios

Líneas 12080-12130: Un módulo de cursor parpadeante que destella en espacio inverso sobre un carácter de la línea 23. La posición del cursor está indicada por el valor de la variable P. El cursor se alterna con una letra de la cadena A$, cadena que se utiliza para introducir el texto.

Línea 12160: Esta línea llama a la parte posterior del programa que incorpora lo que se ha introducido dentro del cuerpo principal del texto. Esto es algo que puede hacer el usuario pulsando ENTER, o que se hace automáticamente cuando la longitud del texto sobrepasa los 40 caracteres y se pulsa un espacio.

Línea 12170: La pulsación de la tecla con el símbolo de «flecha hacia arriba», situada en la fila superior de las teclas del teclado, llama a un módulo posterior que permite que se puedan llevar a cabo una serie de funciones de edición sobre la parte principal del texto.

Líneas 12200-12230: La pulsación del carácter «\» provoca el movimiento del cursor parpadeante al comienzo de la línea o, si ya está allí, al final de la línea.

Líneas 12260-12290: Supuesto que el cursor parpadeante no está colocado sobre el primer carácter, la pulsación de la tecla DEL provoca la supresión del carácter que haya a la izquierda de dicho cursor. Esto

es algo que se hace fácilmente, mediante la reconstrucción de la cadena A$ sin el carácter que ocupaba la posición P.
Líneas 12320-12350: Estas líneas aceptan cualquier carácter cuyo código esté comprendido entre 32 y 136 (para ver la lista completa consulte el Apéndice de su manual), *excepto* la tecla DEL que es tratada de la forma que se ha mencionado más arriba, y lo hacen parte del texto que se está introduciendo. El nuevo carácter se inserta en el texto, justamente en la posición ocupada por el cursor parpadeante.
Líneas 12380-12390: El movimiento del cursor hacia la izquierda o hacia la derecha, se efectúa simplemente cambiando el valor de P de acuerdo con la tecla de cursor que se haya pulsado. Observe como, al igual que en el capítulo de gráficos, la definición de nuestro propio cursor, nos permite aceptar únicamente lo que deseemos de las teclas de control: las cursoras, de inserción, supresión, etc. Ninguna de estas teclas tiene, en absoluto, ningún efecto automático, ya que caen fuera del intervalo de los caracteres de impresión normal que aceptan los bucles previos. Sin embargo, podemos utilizarlas casi como teclas de función, definiendo el efecto de la pulsación de cualquiera de ellas (todas excepto la tecla ESC, por supuesto, que siempre detendrá el programa).

Comprobación

Necesitará, como en programas anteriores, comenzar a insertar líneas del bucle de control, así que introduzca las líneas siguientes:

```
10040 IF IN=0 THEN GOSUB 11000
10060 GOSUB 12000
```

y lance el programa. Especifique que no desea cargar datos desde la cinta y se encontrará frente a frente con un cursor parpadeante situado al comienzo de una línea de la mitad inferior de la pantalla. Comience a teclear y lo que teclee aparecerá en la pantalla. Debería poder borrar caracteres mediante la tecla DEL, mover el cursor sobre lo que ha tecleado, o hacerlo saltar desde su comienzo hasta su final pulsando «\». La pulsación de ENTER, o de cualquiera de las teclas de cursor, o la introducción de más de dos líneas de caracteres seguidas de un espacio, hará que se termine el programa con un mensaje de error «Undefined line» (Línea no definida).

Módulo 4.3.3: Colocación de material en la parte principal del texto

Los dos módulos que vienen a continuación trabajan conjuntamente para permitir que las líneas introducidas en la parte inferior de

la pantalla sean procesadas y convertidas en parte del cuerpo del texto contenido en TEXTO$, matriz que es capaz de guardar hasta 100 líneas de 32 caracteres cada una. El trabajo en sí es realizado por este módulo, pero los resultados solamente resultarán evidentes con la introducción del siguiente módulo, que es quien hace que se visualice una sección de texto.

Módulo 4.3.3: Líneas 13000 - 13310

```
13000 REM*******************************
13010 REM Insertar linea
13020 REM*******************************
13030 IF a$="* " THEN a$=SPACE$(33)
13040 x=0
13050 WHILE a$<>""
13060 :
13070 :
13080 d=0:WHILE LEN(a$)<34 AND d=0
13090 tt$(x)=LEFT$(a$,LEN(a$)-1)
13100 a$=""
13110 d=1:WEND
13120 :
13130 :
13140 WHILE LEN(a$)>33
13150 FOR i=33 TO 1 STEP -1
13160 IF MID$(a$,i,1)<>" " THEN NEXT i:i
=33
13170 tt$(x)=LEFT$(a$,i-1)
13180 a$=MID$(a$,i+1)
13190 x=x+1
13200 WEND
13210 WEND
13220 x=x+1
13230 :
13240 :
13250 FOR i=ul+x TO lu+x STEP -1
13260 texto$(i)=texto$(i-x)
13270 NEXT i
13280 FOR i=0 TO x-1
13290 texto$(lu+i)=tt$(i)
13300 NEXT i
13310 a$=" ":p=0:ul=ul+x:lu=lu+x
```

Comentarios

Línea 13030: Se trata de una simple medida que se incluye para facilitar la salida del texto a una impresora. El problema surge cuando

se desea imprimir una o varias palabras en el lado derecho de la página. La visualización en pantalla, cuando llegamos a generarla, se compone de líneas de 32 caracteres, en tanto que una impresora estándar trabaja con líneas de 80 caracteres.

Durante la impresión, Editexto envía conjuntamente dos líneas de texto de pantalla para conformar una única línea impresa. Por consiguiente, la introducción de texto en la parte derecha de la pantalla, dará lugar a que sea impreso a mitad de la página o pegado al final de la línea anterior. Un método laborioso para superar esta dificultad es introducir primero una línea en blanco, pulsando ENTER, en el sitio en el que no hay que introducir texto (esto hace que se imprima una línea en blanco en el papel, y asegura que la siguiente línea impresa comienza en el lado derecho de la página); después se teclea una línea completa de espacios, seguida de otra línea de espacios que acaba con la palabra a colocar en el lado derecho (la impresora imprimirá todos los espacios, y la frase final aparecerá en la parte derecha del papel).

Sin embargo, en el programa definitivo puede sustituirse la línea completa de espacios por una línea constituida por un único «*». El módulo que nos ocupa, transformará automáticamente esto en una línea de 33 espacios.

Línea 13040: X es la variable que se utilizará para detectar cuántas líneas de texto se necesitan para que sea añadido el texto nuevo.

Líneas 13050-13210: Cada vez que se añade una línea de caracteres a la parte principal de texto, dicha línea es extraída de A$ (que contiene el nuevo texto introducido por medio del módulo anterior). Este proceso continúa hasta que no queda nada en A$.

Líneas 13080-13110: En primer lugar el texto nuevo es colocado en la matriz provisional TT$. Si hay 33 caracteres o menos, es decir menos de 34 incluyendo el espacio que siempre hay al final, entonces el texto nuevo cabrá en una línea de 32 caracteres. Todo lo que se necesita hacer es transferirlo y entonces borrar A$.

Líneas 13140-13200: Si hay más de 33 caracteres, de forma que el texto nuevo no cabe en una única línea, el procedimiento es buscar hacia atrás (a partir del carácter 33) hasta encontrar el primer espacio que señala el final de una palabra. Ahora puede transferirse la parte de A$ que hay hasta este punto, con lo que nos aseguramos de que ninguna palabra será dividida en dos.

Esta sección de texto se suprime ahora de la primera parte de A$, y el bucle principal es ejecutado otra vez exactamente de la misma forma, excepto que otro texto se coloca en una línea diferente de TT$.

Líneas 13250-13300: La variable LU registra el LUgar donde va a ser insertada la nueva línea en la parte principal del texto (la variable comienza por 0 cuando no hay texto, y normalmente permanece al final

de lo que se ha introducido). Módulos posteriores permitirán al usuario mover el punto de inserción hacia adelante o hacia atrás en la parte principal del texto. El primero de los dos bucles de esta sección comienza por hacer sitio, en el punto de inserción, para el número de líneas nuevas que está registrado en X. Esto se hace moviendo todo, desde el punto de inserción hasta la última línea (UL), según el número de líneas representado por X. El segundo bucle copia el contenido de TT$ en el espacio así creado.

Comprobación

Introduzca una línea suplementaria:

```
14130 RETURN
```

y lance entonces el programa. Cuando aparezca el cursor parpadeante, teclee:

```
AAA[ENTER]
BBB[ENTER]
CCC[ENTER]
DDD[ENTER]
```

Habiendo hecho esto, pulse STOP para terminar el programa. Ahora teclee en modo directo:

```
FOR I=0 TO 5:PRINT TEXTO$(I):NEXT[ENTER]
```

Debería ver una línea en diente de sierra, seguida de las cuatro líneas que acaba de introducir, y seguida de otra línea en diente de sierra que marca el final del texto.

Módulo 4.3.4: Visualización del texto

Este módulo visualiza quince líneas de la parte principal del texto.

Módulo 4.3.4: Líneas 14000 - 14130

```
14000 REM*******************************
14010 REM Visualizar seccion de texto
14020 REM*******************************
14030 cs=lu-7
14040 IF ul-lu<8 THEN cs=ul-15
14050 IF cs<0 THEN cs=0
14060 CLS
```

```
14070 FOR i=cs TO cs+15
14080 PEN 2
14090 PRINT texto$(i)
14100 IF i=lu-1 THEN PEN 1:PRINT ">"
14110 NEXT i
14120 LOCATE 1,23:PEN 2:PRINT a$:PEN 1
14130 RETURN
```

Comentarios
Líneas 14030-14050: La variable CS representa la línea de comienzo de la sección a presentar (Comienzo de Sección). Se calcula de forma que esté normalmente ocho líneas por delante del punto de inserción actual. Si el punto de inserción está cerca del final del texto, digamos al final del todo, significaría que sólo se visualizarían ocho líneas, por lo que la línea 14040 mueve el punto de comienzo más atrás. Por último, si hay menos de 15 líneas, o el punto de inserción está al principio, entonces el punto de comienzo estará ahora antes del principio del texto, por lo que dicho punto se incrementa hasta 0.
Líneas 14070-14110: Ahora se presentan las 15 líneas. Todas las líneas terminan con un punto y coma para impedir que la posición de impresión baje una línea. De lo contrario, una línea completa de 40 caracteres (la primera y la última línea, que están formadas por símbolos gráficos) enviaría el cursor a la línea siguiente, dejando por tanto una línea en blanco al bajar. Para líneas de menos de 40 caracteres, el efecto del punto y coma es cancelado mediante un PRINT vacío. El único otro refinamiento es que se marca el punto de inserción mediante un signo «>» (cualquiera de las funciones de edición tales como añadir líneas, copiarlas o suprimir actuarán sobre la línea que hay por debajo del marcador).
Línea 14120: Por último, al tiempo que fue borrada la pantalla, cualquier cosa que haya en A$ es vuelta a presentar en la parte inferior de la pantalla.

Comprobación
Haga la misma comprobación que realizó con el último módulo, con la única diferencia de que en vez de tener que introducir una orden especial para ver lo que ha introducido, cada línea debería colocarse en el texto principal cuando pulse ENTER.

Módulo 4.3.5: Edición del texto principal

Habiéndonos proporcionado la capacidad de editar el texto nuevo conforme va siendo introducido, ahora necesitamos suministrar algún tipo de control sobre el propio texto principal. Este módulo proporciona

al usuario la capacidad de mover el punto de inserción a lo largo del texto, de copiar líneas desde el texto a la parte inferior de la pantalla para efectuar posibles alteraciones, y de suprimir líneas existentes.

Módulo 4.3.5: Líneas 15000 - 15320

```
15000 REM*******************************
15010 REM Modo edicion
15020 REM*******************************
15030 WHILE 1
15040 p2=lu-cs
15050 :
15060 :
15070 t1$="":WHILE t1$=""
15080 LOCATE 1,p2+1:PRINT " ":FOR i=1 TO
 5:NEXT i
15090 LOCATE 1,p2+1:PRINT ">":FOR i=1 TO
 20:NEXT i
15100 t1$=LOWER$(INKEY$)
15110 WEND
15120 :
15130 :
15140 lu=lu+(t1$=CHR$(240))+10*(t1$="s")
:IF lu<1 THEN lu=1
15150 lu=lu-(t1$=CHR$(241))-10*(t1$="b")
:IF lu>ul THEN lu=ul
15160 IF t1$=CHR$(13) THEN t$="":RETURN
15170 :
15180 :
15190 WHILE lu<ul AND t1$=CHR$(127)
15200 FOR i=lu TO ul:texto$(i)=texto$(i+
1):NEXT i
15210 ul=ul-1
15220 t1$="":WEND
15230 :
15240 :
15250 IF lu<ul AND t1$="c" THEN a$=texto
$(lu)+" "
15260 IF t1$="i" OR t1$="p" THEN GOSUB 1
7000
15270 IF t1$="g" THEN GOSUB 18000
15280 IF t1$="f" THEN GOSUB 16000
15290 :
15300 :
15310 GOSUB 14000
15320 WEND
```

Comentarios

Línea 15040: Para los fines de este módulo, se almacenará en la variable P2 el número de línea de pantalla donde se encuentra el símbolo «>» (símbolo que indica el punto de inserción actual en el texto).

Líneas 15070-15110: Se hace parpadear al cursor «>», para indicar que el programa está en el modo de edición.

Líneas 15140-15150: El cursor de edición puede ser movido hacia arriba o hacia abajo a lo largo del texto (aunque únicamente en el modo de edición). La pulsación de las teclas cursoras de flecha hacia arriba o hacia abajo hace que el cursor se mueva una línea en la dirección indicada. La pulsación de las teclas «S» (de Subir) o «B» (de Bajar) hace que el cursor suba o baje 10 líneas. Observe que, en la práctica, y a menos que lleguemos al principio o al final del fichero, lo que se mueve es el texto en torno al cursor y no el cursor en sí.

Línea 15160: La pulsación de ENTER concluye el modo de edición.

Líneas 15190-15220: La pulsación de la tecla DEL, en el modo de edición, suprime la línea de texto que hay bajo el cursor de edición.

Línea 15250: La pulsación de «C», en el modo de edición, copia la línea que hay bajo el cursor de edición en la parte inferior de la pantalla, de forma que puede ser manipulada como si se tratara de texto nuevo.

Línea 15260: La pulsación de «I» o «P», en el modo de edición, llama al módulo posterior que envía el texto a la Impresora o a la Pantalla en modo de 80 columnas.

Línea 15270: La pulsación de «G», en el modo de edición, llama al módulo posterior que Graba el texto en cinta.

Línea 15280: La pulsación de «F» (Formatear), en el modo de edición, llama al módulo posterior que ordena el texto e intenta uniformar tanto como sea posible la longitud de las líneas.

Comprobación

Una vez introducido el módulo, efectúe la misma comprobación que realizó con los módulos anteriores, insertando cuatro grupos de letras.

1) Ahora pulse «^», y el cursor de edición debería comenzar a parpadear.
2) Experimente los movimientos hacia arriba y hacia abajo del cursor de edición mediante las teclas cursoras. La pulsación de «S» o «B» debería mover el cursor a la parte superior o a la inferior del texto, ya que hay menos de 10 líneas de texto.
3) Mueva el cursor de edición a la línea que hay por encima de la última línea de texto. Pulse «C» y el grupo «DDD» debería aparecer copiado en la parte inferior de la pantalla.

4) Pulse la tecla DEL, y la línea «DDD» debería desaparecer del texto principal.
5) Mueva el cursor de edición a la parte superior del texto.
6) Pulse ENTER para finalizar el modo de edición.
7) Cuando el cursor parpadeante retorne a la parte inferior de la pantalla pulse ENTER de nuevo. Debería ver el grupo «DDD» insertado al comienzo del texto.

Módulo 4.3.6: Formateado del texto

Otra posibilidad útil es la de ordenar la parte principal de texto que se ha introducido. Esto es necesario, pues una de las principales ventajas de la edición de textos, o de su hermano mayor: el procesamiento de palabras, es la capacidad de tomar un texto existente y extraer o insertarle frases nuevas. Muy a menudo este proceso deja el texto con una apariencia desigual y descuidada, por lo que la mayoría de los procesadores de palabras que no ordenan automáticamente el texto conforme se va introduciendo, permiten al usuario la llamada a una función de «formato». En su forma más simple, la función del formateo es examinar cada línea y determinar si la primera palabra de la línea siguiente se podría añadir al final de la línea examinada (y en caso positivo efectuar el movimiento consiguiente). El resultado es un texto de apariencia más cuidada y elegante, incluso aún cuando no se añadan espacios en blanco para hacer que todas las líneas terminen exactamente en el mismo punto (proceso denominado justificación o alineación de márgenes). El módulo que nos ocupa realiza la función de formateo sobre el texto guardado en TEXTO$, utilizando para ello las técnicas de corte en rebanadas (disección) de cadenas expuestas por primera vez en el Capítulo 1.

Módulo 4.3.6: Líneas 16000 - 16310

```
16000 REM*******************************
16010 REM Formatear linea
16020 REM*******************************
16030 FOR i=1 TO ul-2
16040 d=0:WHILE texto$(i)<>"" AND texto$
(i+1)<>"" AND texto$(i)<>SPACE$(32) AND
texto$(i+1)<>SPACE$(32) AND d=0
16050 :
16060 :
16070 es=32-LEN(texto$(i))
16080 plb=INSTR(texto$(i+1)," ")
16090 IF plb=0 THEN plb=LEN(texto$(i+1))
+1
```

```
16100 IF plb>es THEN d=1
16110 :
16120 :
16130 d2=0:WHILE es>=plb AND es<=LEN(tex
to$(i+1)) AND d2=0
16140 texto$(i)=texto$(i)+" "+LEFT$(text
o$(i+1),plb-1)
16150 texto$(i+1)=MID$(texto$(i+1),plb+1
)
16160 d2=1:WEND
16170 :
16180 :
16190 es=32-LEN(texto$(i))
16200 d2=0:WHILE LEN(texto$(i+1))<es AND
 d=0 AND d2=0
16210 texto$(i)=texto$(i)+" "+texto$(i+1
)
16220 FOR j=i+1 TO ul
16230 texto$(j)=texto$(j+1)
16240 NEXT j
16250 ul=ul-1:lu=lu-1
16260 d2=1:WEND
16270 :
16280 :
16290 WEND
16300 NEXT i
16310 RETURN
```

Comentarios

Líneas 16030-16300: El proceso de formateo comienza por la primera línea, y se abre paso por el texto línea a línea.

Líneas 16040-16290: La operación solamente se realiza sobre aquellas líneas que contienen algo. Al final de un párrafo, por ejemplo, puede haber muy bien una línea con menos de 32 caracteres, y no desearemos ver cómo se enlaza el comienzo del párrafo siguiente con esta línea. Esto puede ser superado por el usuario, mediante la introducción de una línea vacía tras el final del párrafo. El módulo de formato, ni añadirá esta línea vacía al final de la línea anterior, ni copiará ninguna palabra de la línea siguiente sobre dicha línea.

Línea 16070: La variable ES se utiliza para almacenar la cantidad de espacio libre que queda al final de la línea que se está examinando.

Línea 16080: PLB se utiliza para almacenar la longitud de la primera palabra de la línea siguiente, según queda indicada por la presencia de un espacio.

Línea 16090: Si no hay ningún espacio en la línea siguiente, la longitud de PLB se fija a la longitud de la línea completa.

Línea 16100: Si la longitud de la primera palabra de la línea siguiente es mayor que el espacio disponible al final de la línea examinada, no tiene ningún sentido continuar con dicha línea, por lo que se termina el bucle WHILE, y el bucle FOR pasa a la línea siguiente del texto.
Líneas 16130-16160: Estas líneas son llamadas a la acción en caso de que haya suficiente espacio al final de la línea examinada para la palabra siguiente, aunque no para la totalidad de la línea siguiente. Su efecto es copiar la palabra al final de la línea, y suprimirla del comienzo de la siguiente.
Líneas 16190-16260: Es perfectamente posible que la línea examinada tenga suficiente espacio como para guardar la totalidad de la línea siguiente. En este caso no se trata simplemente de una cuestión de copiar la línea siguiente en la examinada: la totalidad de la matriz tiene que colapsarse por una línea para llenar el espacio creado, en caso contrario el programa será confundido por la instrucción, mencionada anteriormente, de que las líneas vacías deben quedar completamente solas.

Comprobación

Ejecute la misma comprobación anterior, introduciendo los cuatro grupos de letras. Cuando estén visualizados formando parte del texto principal, pulse «^» para meter el modo de edición, y entonces pulse «F». Tras una pausa (que puede ser bastante grande cuando introduzca grandes cantidades de texto) se volverá a visualizar el texto principal, pero esta vez los cuatro grupos de letras deberían estar juntos en una sola línea.

Módulo 4.3.7: Ficheros de datos

Un módulo estándar de fichero de datos. Una cosa a tener en cuenta es el uso de LINE INPUT, que asegura que los textos que contienen comas y otros caracteres, a los que un INPUT se hubiera negado, son recogidos correctamente.

Por otra parte, el módulo parece haber sacado a la luz un error sumamente raro que padecen, al menos, algunos 464. El error toma la forma de una corrupción o deterioro, totalmente invisible para el usuario, de la cadena FI$ durante el recorrido del módulo; dicho deterioro hace que el 464 no reconozca el nombre del fichero de la cinta como igual al introducido, todo ello a pesar de que cuando se visualizan los dos resultan ser idénticos. En nuestra propia versión del programa, la línea siguiente:

```
19075 FI$=FI$
```

por ridículo que parezca, supera el problema, recordándole al ordenador el verdadero valor de la cadena. Si su programa funciona sin este añadido (como debiera ser) puede ignorar todo este comentario.

Módulo 4.3.7: Líneas 18000 - 19150

```
18000 REM*******************************
18010 REM Grabar en cinta
18020 REM*******************************
18030 q$="":WHILE LOWER$(q$)<>"s"
18040 CLS:INPUT "Nombre del fichero:",fi
$
18050 PRINT "El fichero que se va a grab
ar se llama ";fi$
18060 INPUT "Es correcto (s/n)";q$
18070 WEND
18080 OPENOUT fi$
18090 PRINT #9,lu
18100 PRINT #9,ul
18110 FOR i=0 TO ul
18120 PRINT #9,texto$(i)
18130 NEXT i
18140 CLOSEOUT
18150 RETURN
19000 REM*******************************
19010 REM Cargar desde la cinta
19020 REM*******************************
19030 q$="":WHILE LOWER$(q$)<>"s"
19040 PRINT:INPUT "Nombre del fichero:",
fi$
19050 PRINT "El fichero que se va a carg
ar se llama ";fi$
19060 INPUT "Es correcto (s/n)";q$
19070 WEND
19080 OPENIN fi$
19090 INPUT #9,lu
19100 INPUT #9,ul
19110 FOR i=0 TO ul
19120 LINE INPUT #9,texto$(i)
19130 NEXT i
19140 CLOSEIN
19150 RETURN
```

Módulo 4.3.8: Módulo de control

Un módulo estándar de control.

Módulo 4.3.8: Líneas 10000 - 10060

```
10000 REM*******************************
10010 REM Bucle de control
10020 REM*******************************
10030 MODE 1
10040 IF in=0 THEN GOSUB 11000
10050 GOSUB 14000
10060 GOTO 12000
```

Módulo 4.3.9: Impresión del texto

Si posee una impresora (y un editor de textos no le va a resultar muy útil si no la tiene), entonces este módulo le permitirá coger lo que ha compuesto sobre la pantalla y ponerlo sobre papel. Además, el módulo hace posible que el usuario pueda ver primero el texto en pantalla, en el modo de 80 columnas, asegurándose así de que la distribución del texto es tal, que se obtendrá un resultado satisfactorio en la impresora.

Módulo 4.3.9: Líneas 17000 - 17250

```
17000 REM*******************************
17010 REM Salida por impresora
17020 REM*******************************
17030 canal=8
17040 IF t1$="p" THEN canal=0:PEN 1:MODE
 2
17050 x=1
17060 WHILE x<ul
17070 d=0
17080 :
17090 :
17100 IF texto$(x)="" THEN PRINT #canal:
d=1
17110 IF d=0 THEN PRINT #canal,SPACE$(8)
;texto$(x);" ";
17120 x=x+1
17130 :
17140 :
17150 WHILE x<ul AND d=0
17160 PRINT #canal,texto$(x)
17170 IF texto$(x)="" THEN PRINT #canal
17180 x=x+1
17185 IF canal=0 THEN IF INKEY$="" THEN
GOTO 17185
```

```
17190 d=1:WEND
17200 :
17210 :
17220 WEND
17230 PRINT #canal
17240 IF canal=0 THEN IF INKEY$="" THEN
GOTO 17240
17250 MODE 1:RETURN
```

Comentarios

Líneas 17030-17040: Este módulo es llamado si el usuario desea ver el texto en una impresora o en pantalla en el modo de 80 columnas. La diferencia será que, si el usuario desea visualizarlo en pantalla, entonces la salida del programa irá al canal 0 (la pantalla) y se activará el modo 2 (MODE 2).

Línea 17050: La variable X se utiliza para registrar el avance del módulo a lo largo de las líneas de TEXTO$.

Línea 17070: La variable D se utilizará para saltar sobre secciones del módulo, cuando no hay que hacer nada más sobre una línea particular del texto.

Línea 17100: Igual que durante el formateo, no se hace ningún intento de interferir con las líneas vacías; cuando se encuentra una, se emplea una sentencia PRINT # vacía para bajar una línea la posición de impresión. La variable D se pone a 1 para indicar que se ha terminado una línea de texto.

Línea 17110: Si la línea en curso no está vacía, entonces se imprime como la primera mitad de una línea de 64 caracteres, seguida de un punto y coma. Tome nota del punto y coma, que asegura que los siguientes caracteres irán a continuación.

Líneas 17150-17190: Se imprime la segunda mitad de la línea. Si era una línea vacía, se utiliza un PRINT # extra para bajar un espacio el punto de impresión. Si la salida se está enviando a la pantalla, entonces se inserta una pausa tras cada línea impresa hasta que el usuario pulsa una tecla. Esto último permite visualizar documentos que ocupan más de una única pantalla.

Línea 17240: Una vez más, si se está utilizando la pantalla para visualizar el texto, se espera a que se pulse otra tecla cuando se ha terminado el documento. Esto es para impedir que el usuario se pase la última línea involuntariamente y pierda la visualización del documento.

Comprobación

Si tiene una impresora, asegúrese de que está conectada adecuadamente y enciéndala, introduzca entonces algún texto. Pulse «^» para meter el modo de edición y finalmente pulse «l». El texto debería

ser sacado por la impresora. Si esta prueba se realiza con éxito, el programa debería estar listo para ser usado. De cualquier manera, para ayudarle a cogerle el tranquillo al programa, más abajo se da una tabla de las diversas órdenes «monotecla» que debe utilizar.

Editexto: Tabla de órdenes «monotecla»

En modo de introducción de texto

ENTER	Introduce nuevo texto en el texto principal. Mueve el cursor al principio o al final de la línea.
DEL	Suprime el carácter de la izquierda del cursor.
TECLAS CURSORAS	Mueven el cursor a la izquierda o a la derecha. Introduce el modo de edición.

En modo de edición

ENTER	Termina el modo de edición.
TECLAS CURSORAS	Mueven el cursor de edición una línea hacia arriba o hacia abajo.
S o B	Mueven el cursor de edición diez líneas hacia arriba o hacia abajo.
DEL	Suprime la línea que hay inmediatamente debajo del cursor de edición.
C	Copia la línea que hay inmediatamente debajo del cursor de edición.
I	Imprime el texto principal vigente.
G	Graba en cinta el texto principal vigente.
F	Formatea el texto principal.
P	Presenta el texto en formato de 80 columnas.

PROGRAMA 4.4: Multirrespuesta

Función del programa

En el último programa de este capítulo pasaremos a tocar los respetables temas culturales, en busca de un poco de entretenimiento. No estoy completamente seguro de cuánto puede aprender,

utilizando este programa, sobre el tema que haya escogido, pero es divertido y crea adicción a la respuesta de preguntas. No sólo eso, el programa es una estrella por derecho propio, ya que una versión previa de Multirrespuesta fue el primer programa (al menos de acuerdo con los comunicados de prensa) en regular un concurso para radioyentes en Gran Bretaña.

Multirrespuesta, como sugiere su nombre, es un programa de usos múltiples, que en cierto momento puede ser un profesor particular de idiomas, y que unos minutos más tarde puede estar interrogándole sobre enrevesadas cuestiones de la historia del siglo XIX. Todo esto lo hace generando tests de respuesta aleatoria (de los del tipo que cada vez se usan más en los exámenes multitudinarios), estableciendo una pregunta y proporcionando cinco respuestas posibles, de las cuales solamente una es la correcta. Simultáneamente se va guardando una puntuación, que proporciona una evaluación de los conocimientos del usuario sobre el tema en cuestión. Obviamente, el trabajo más pesado corre a cargo del programador, ya que no sólo tiene que introducir el programa, está también el asuntillo de meter un conjunto suficientemente grande de preguntas como para asegurar que los tests sean significativos.

Módulo 4.4.1: Inicialización

Como en todos los programas de usos múltiples de este capítulo, este módulo incluye los medios para describir el tipo de fichero que se va a manejar en la sesión de ese momento.

Módulo 4.4.1: Líneas 10000 - 10100

```
10000 REM*******************************
10010 REM Inicializacion
10020 REM*******************************
10030 DIM enca$(1),qu(4),lista$(499,1),q
tipo(9),ntipo(1,9)
10040 correctas=0:total=0:pr=0
10050 MODE 1
10060 PEN 2:PRINT TAB(13);"MULTIRRESPUES
TA":PRINT TAB(13);"=============="
10070 WINDOW 1,40,4,25
10080 PEN 1
10090 INPUT "Desea cargar desde al cinta
 (s/n)";q$
10100 IF LOWER$(q$)="s" THEN GOSUB 23000
 ELSE GOSUB 24000
```

Comentarios

Línea 10030: La matriz ENCA$ se utilizará para registrar los encabezamientos o títulos generales dados por el usuario a las preguntas y a las respuestas, QU se utilizará en la preparación de tests aleatorios, LISTA$ guardará el fichero principal de preguntas y respuestas, QTIPO$ guardará los nombres de los tipos que se pueden asignar a las preguntas y a las respuestas y NTIPO almacenará el número que hay de cada tipo en el fichero principal.

```
                MULTIRRESPUESTA
                ===============

ACONTECIMIENTO:
Comienzo de la II Guerra Mundial

FECHA:
1 ) 1937
2 ) 1941
3 ) 1938
4 ) 1940
5 ) 1939

Cual es la respuesta correcta (1 a 5)? 5

*************
*           *
* CORRECTO! *
*           *
*************

Mas preguntas (s/n)?
```

Fig. 4.4 Parte de un test preparado por Multirrespuesta.

Módulo 4.4.2: La estructura del test

Multirrespuesta, como ya hemos hecho observar, prepara tests, es decir, pregunta cuestiones y presenta respuestas posibles. Estas líneas le permiten al usuario dar un encabezamiento general para las preguntas y las respuestas. Si el programa se fuera a utilizar, por ejemplo, para aprender inglés, entonces podría llamar «PALABRA EN ESPAÑOL» a la pregunta y «PALABRA INGLESA EQUIVALENTE» a la respuesta.

Módulo 4.4.2: Líneas 24000 - 24130

```
24000 REM*******************************
24010 REM Estructura del test
24020 REM*******************************
24030 q$="":WHILE LOWER$(q$)<>"s"
24040 CLS
24050 PRINT "Estructura del test:":PRINT
24060 INPUT "Encabezamiento para la resp
uesta:",enca$(0)
24070 INPUT "Encabezamiento para la preg
unta:",enca$(1)
24080 PRINT:INPUT "Es correcto (s/n)";q$
24090 WEND
24100 qtipo$(0)="Ningun tipo en especial
"
24110 ntipo=1
24120 GOSUB 12000
24130 RETURN
```

Módulo 4.4.3: Mensajes de error

Un módulo estándar para presentar mensajes de error proporcionados por otras secciones del programa.

Módulo 4.4.3: Líneas 25000 - 25070

```
25000 REM*******************************
25010 REM Error
25020 REM*******************************
25030 PRINT:PRINT x$
25040 SOUND 1,1000,100
25050 FOR i=1 TO 1500
25060 NEXT i
25070 RETURN
```

Módulo 4.4.4: El menú

Un módulo de menú estándar.

Módulo 4.4.4: Líneas 11000 - 11210

```
11000 REM*******************************
11010 REM Menu principal
11020 REM*******************************
11030 WHILE z<>7
```

```
11040 CLS
11050 PRINT "ORDENES DISPONIBLES:"
11060 PRINT
11070 PRINT "1) Introducir nuevas pregun
tas/respues-    tas."
11080 PRINT "2) Introducir nuevos tipos.
"
11090 PRINT "3) Buscar/Borrar."
11100 PRINT "4) Generar preguntas."
11110 PRINT "5) Mostrar la puntuacion o
ponerla a ce-   ro."
11120 PRINT "6) Grabar datos en cinta."
11130 PRINT "7) Terminar."
11140 PRINT
11150 INPUT "Cual elige: ",z
11160 IF z>2 AND z<7 AND pr=0 THEN x$="
*NO SE HAN INTRODUCIDO DATOS TODAVIA*":G
OSUB 25000:z=0
11170 ON z GOSUB 13000,12000,20000,17000
,19000,22000
11180 WEND
11190 CLS
11200 LOCATE 14,11:PRINT "AULA CERRADA"
11210 END
```

Módulo 4.4.5: Preparación de los tipos de pregunta

Conforme vaya introduciendo módulos posteriores, descubrirá que Multirrespuesta prevé dos niveles diferentes de dificultad para los test que prepara. Esto se hace en base a tipos de pregunta. Volviendo al ejemplo de la utilización del programa como profesor de inglés, es evidente que resulta posible dividir las palabras a presentar en diferentes grupos gramaticales, como verbos, sustantivos, adjetivos, etc. Si se presenta una palabra española que es un verbo, y de las cinco respuestas posibles en inglés solamente una de ellas es un verbo, entonces el test será mucho más fácil que si las cinco respuestas posibles fueran verbos.

El objetivo de este módulo es permitirle al usuario la definición de hasta 10 tipos diferentes entre los que se puedan incluir las preguntas o las respuestas, junto con la posibilidad de adjuntar uno de los tipos a una pregunta conforme esta se teclea. Posteriormente, cuando Multirrespuesta pase a preparar un test, preguntará si el usuario desea que las posibles respuestas a las preguntas sean extraídas solamente de entre las del mismo tipo que la de la respuesta correcta, o de entre el conjunto total de respuestas.

Módulo 4.4.5: Líneas 12000 - 12150

```
12000 REM*******************************
12010 REM Nuevos tipos
12020 REM*******************************
12030 q$="":WHILE 1
12040 CLS
12050 PRINT "Tipos:":PRINT
12060 PRINT "Tipos que hay hasta ahora:"
:PRINT
12070 FOR i=0 TO ntipo-1
12080 PRINT i+1;") ";qtipo$(i)
12090 NEXT i
12100 IF ntipo=10 THEN x$="** NO SE PUED
EN INTRODUCIR MAS TIPOS **":GOSUB 25000:
RETURN
12110 PRINT:INPUT "Introduzca nuevo tipo
 ('\' para volver  al menu):";q$
12120 IF q$="\" THEN RETURN
12130 qtipo$(ntipo)=q$
12140 ntipo=ntipo+1
12150 WEND
```

Comprobación

Ahora debería estar en posición de lanzar el programa e introducir hasta 10 tipos de preguntas. Para confirmar que los tipos han sido aceptados puede detener el programa y teclear:

FOR I=0 TO 9:PRINT QTIPO$(I):NEXT I[ENTER]

Módulo 4.4.6: Búsqueda binaria

Un módulo estándar de búsqueda, trabajando según orden alfabético sobre la base de las respuestas a las preguntas. Observe que, como descubrirá cuando introduzca dentro de un momento el módulo de nuevas preguntas/respuestas, cada respuesta va precedida de un carácter (0-9) que indica cuál es su tipo. Por consiguiente, el fichero será ordenado atendiendo al tipo en primer lugar, y dentro de cada tipo atendiendo al orden alfabético de las respuestas.

Módulo 4.4.6: Líneas 14000 - 14110

```
14000 REM*******************************
14010 REM Busqueda binaria
14020 REM*******************************
```

```
14030 IF pr=0 THEN bb=0:RETURN
14040 po=INT(LOG(pr)/LOG(2)):bb=2^po-1
14050 FOR i=po TO 0 STEP -1
14060 bb=bb+2^i*((lista$(bb,0)>t1$)-(lis
ta$(bb,0)<t1$))
14070 IF bb<0 THEN bb=0
14080 IF bb>pr-1 THEN bb=pr-1
14090 NEXT i
14100 IF lista$(bb,0)<t1$ THEN bb=bb+1
14110 RETURN
```

Módulo 4.4.7: Inserción de una pregunta/respuesta

Un módulo estándar de inserción.

Módulo 4.4.7: Líneas 15000 - 15110

```
15000 REM*******************************
15010 REM Insertar entrada
15020 REM*******************************
15030 FOR i=pr TO bb+1 STEP -1
15040 lista$(i,0)=lista$(i-1,0)
15050 lista$(i,1)=lista$(i-1,1)
15060 NEXT i
15070 lista$(bb,0)=t1$
15080 lista$(bb,1)=t2$
15090 pr=pr+1
15100 GOSUB 16000
15110 RETURN
```

Módulo 4.4.8: Seguimiento de los tipos

Ya hemos introducido el módulo que permite registrar los tipos, pero también necesitamos suministrar al programa la capacidad de registrar cuántos de cada tipo hay en el fichero, y dónde comienza cada grupo de tipos.

Del registro de cuántos hay de cada tipo se ocupa el módulo de introducción, que simplemente suma 1 al elemento pertinente de la matriz NTIPO. Este módulo es llamado siempre que se hace una nueva entrada o una supresión. Su propósito es registrar en el otro lado de NTIPO los totales acumulados de los tipos 0 a 9. En determinado momento, las respuestas se dispondrán según orden de tipos dentro del fichero, por lo que el conocimiento del número de elementos que hay en los tipos 0 a 2, por ejemplo, nos dirá dónde comienzan los elementos que caen bajo el grupo 3.

Módulo 4.4.8: Líneas 16000 - 16080

```
16000 REM*******************************
16010 REM Actualizacion
16020 REM*******************************
16030 su=0
16040 FOR i=0 TO 9
16050 ntipo(1,i)=su
16060 su=su+ntipo(0,i)
16070 NEXT i
16080 RETURN
```

Módulo 4.4.9: Introducción de una pregunta/respuesta nueva

Un módulo sencillo que permite al usuario la introducción de una nueva pregunta y respuesta, después de lo cual le añade el tipo.

Módulo 4.4.9: Líneas 13000 - 13300

```
13000 REM*******************************
13010 REM Nuevas preguntas/respuestas
13020 REM*******************************
13030 WHILE 1
13040 CLS
13050 PRINT "Nuevas preguntas/respuestas
:":PRINT
13060 IF pr=500 THEN LET x$="     **** NO
 HAY SITIO PARA MAS *****":GOSUB 25000:R
ETURN
13070 PRINT "Introduzca la pregunta/resp
uesta especificada."
13080 PRINT "'\' para volver al menu."
13090 PRINT
13100 PRINT enca$(0);
13110 INPUT ":";t1$
13120 IF t1$="\" THEN RETURN
13130 PRINT enca$(1);
13140 INPUT ":";t2$
13150 IF t2$="\" THEN RETURN
13160 PRINT:PRINT "Tipos:":PRINT
13170 FOR i=0 TO ntipo-1
13180 PRINT i+1;") ";qtipo$(i)
13190 NEXT i
13200 PRINT:INPUT "Introduzca numero del
 tipo:",t3
13210 t3=t3-1
```

```
13220 IF t3<0 OR t3>ntipo THEN t3=0
13230 PRINT:INPUT "Es correcto (s/n)";q$
13240 WHILE LOWER$(q$)="s"
13250 ntipo(0,t3)=ntipo(0,t3)+1
13260 t1$=MID$(STR$(t3)+t1$,2)
13270 GOSUB 14000
13280 GOSUB 15000
13290 q$="":WEND
13300 WEND
```

Comentarios
Línea 13250: Se incrementa en 1 el elemento de la matriz NTIPO que representa el tipo especificado por la pregunta y respuesta actual. Es esta matriz, como hemos visto, la que es investigada por el módulo de actualización para registrar dónde comienza cada grupo dentro de la matriz.

Comprobación

Ahora debería poder ejecutar el programa, especificar tipos, y llamar entonces a la opción 1 del menú para comenzar a meter preguntas y respuestas. Para verificar que se están recibiendo correctamente, introduzca los datos siguientes:

PREGUNTA	RESPUESTA	TIPO
P111	R111	3
P222	R222	2
P333	R333	1

y abandone entonces el programa. Ahora teclee:

FOR I=0 TO 2:FOR J=0 TO 1:PRINT LISTA$(I,J):NEXT J:NEXT I[ENTER]

y debería ver:

```
0R333
P333
1R222
P222
2R111
P111
```

Módulo 4.4.10: Almacenamiento de los datos

Ahora que ya podemos introducir datos, ha llegado el momento de poner los dos módulos estándar que los almacenarán y llamarán.

```
22000 REM*******************************
22010 REM Grabar en cinta
22020 REM*******************************
22030 CLS
22040 PRINT "Grabar datos:":PRINT
22050 INPUT "Nombre del fichero:",fi$
22060 OPENOUT fi$
22070 PRINT #9,pr
22080 PRINT #9,ntipo
22090 FOR i=0 TO 1
22100 PRINT #9,enca$(i)
22110 FOR j=0 TO ntipo-1
22120 PRINT #9,ntipo(i,j)
22130 NEXT j
22140 FOR j=0 TO pr-1
22150 PRINT #9,lista$(j,i)
22160 NEXT j
22170 NEXT i
22180 FOR i=0 TO ntipo-1
22190 PRINT #9,qtipo$(i)
22200 NEXT i
22210 CLOSEOUT
22220 RETURN
23000 REM*******************************
23010 REM Cargar desde la cinta
23020 REM*******************************
23030 CLS
23040 PRINT "Cargar datos:":PRINT
23050 INPUT "Nombre del fichero:",fi$
23060 OPENIN fi$
23070 INPUT #9,pr
23080 INPUT #9,ntipo
23090 FOR i=0 TO 1
23100 INPUT #9,enca$(i)
23110 FOR j=0 TO ntipo-1
23120 INPUT #9,ntipo(i,j)
23130 NEXT j
23140 FOR j=0 TO pr-1
23150 INPUT #9,lista$(j,i)
23160 NEXT j
23170 NEXT i
23180 FOR i=0 TO ntipo-1
23190 INPUT #9,qtipo$(i)
23200 NEXT i
23210 CLOSEIN
23220 RETURN
```

Módulo 4.4.11: Búsqueda por parte del usuario

Un módulo sencillo que permite al usuario la búsqueda hacia adelante y hacia atrás por el fichero, visualizando y, una vez se haya introducido el módulo siguiente, borrando preguntas/respuestas.

Módulo 4.4.11: Líneas 20000 - 20220

```
20000 REM*******************************
20010 REM Busqueda
20020 REM*******************************
20030 bb=0
20040 t1$="":WHILE t1$<>"\" AND pr>0
20050 CLS:PRINT "Busqueda:":PRINT
20060 PRINT "Numero de la pregunta/respu
esta:";bb+1
20070 PRINT enca$(0);":";MID$(lista$(bb,
0),2)
20080 PRINT enca$(1);":";lista$(bb,1)
20090 PRINT "Tipo:";qtipo$(VAL(LEFT$(lis
ta$(bb,0),1)))
20100 PRINT:PRINT "Ordenes disponibles:"
:PRINT
20110 PEN 2:PRINT "ENTER";:PEN 1:PRINT "
 para ver la siguiente."
20120 PRINT "'#' y un numero para mover
el puntero."
20130 PRINT "'s' para suprimirla."
20140 PRINT "'\' para volver al menu."
20150 PRINT:INPUT "Cual desea";t1$
20160 IF t1$="" THEN t1$="#1"
20170 IF LOWER$(t1$)="s" THEN GOSUB 2100
0
20180 IF LEFT$(t1$,1)="#" THEN bb=bb+VAL
(MID$(t1$,2))
20190 IF bb>pr-1 THEN bb=pr-1
20200 IF bb<0 THEN bb=0
20210 WEND
20220 RETURN
```

Comprobación

Lance el programa y llame a los datos que ha almacenado en cinta. Utilizando la opción 3 del menú, debería poder recorrer hacia adelante y hacia atrás las preguntas/respuestas.

Módulo 4.4.12: Supresión de una pregunta/respuesta

Un módulo estándar de anulación o supresión.

Módulo 4.4.12: Líneas 21000 - 21100

```
21000 REM*******************************
21010 REM Suprimir
21020 REM*******************************
21030 ntipo(0,VAL(LEFT$(lista$(bb,0),1))
)=ntipo(0,VAL(LEFT$(lista$(bb,0),1)))-1
21040 FOR i=bb TO pr-2
21050 lista$(i,0)=lista$(i+1,0)
21060 lista$(i,1)=lista$(i+1,1)
21070 NEXT i
21080 pr=pr-1
21090 GOSUB 16000
21100 RETURN
```

Comprobación

Siga el procedimiento del módulo anterior, pero pulse «S» frente a una u otra de las preguntas/respuestas. Debería encontrar que la pregunta/respuesta correspondiente ha sido suprimida.

Módulo 4.4.13: Preparando las preguntas

Ahora pasamos a los módulos que constituyen la novedad de Multirrespuesta al preparar los tests multielección. Este módulo se ocupa de la parte visible del proceso: la presentación de las preguntas y de las posibles respuestas, y de la elección del usuario respecto a cuál es la respuesta correcta.

Módulo 4.4.13: Líneas 17000 - 17430

```
17000 REM*******************************
17010 REM Preguntas
17020 REM*******************************
17030 CLS
17040 PRINT "Preguntas:":PRINT
17050 PRINT "Desea que las respuestas se
an extraidas";
17060 PRINT "de uno de los tipos (mas di
ficil), o de"
```

```
17070 PRINT "entre todos los tipos (mas
facil)?"
17080 PRINT:PRINT "1) Solo de uno de los
 tipos."
17090 PRINT "2) De entre todos los tipos
."
17100 PRINT:INPUT "Cual de las dos opcio
nes:",rq
17110 IF rq<1 OR rq>2 THEN rq=2
17120 q$="":WHILE LOWER$(q$)<>"n"
17130 GOSUB 18000
17140 CLS
17150 PRINT enca$(1);":";lista$(qu(qpos)
,1)
17160 PRINT:PRINT enca$(0);":"
17170 FOR i=0 TO 4
17180 PRINT i+1;") ";MID$(lista$(qu(i),0
),2)
17190 NEXT i
17200 cc=0:WHILE cc<1 OR cc>5
17210 PRINT:INPUT "Cual es la respuesta
correcta (1 a 5)";cc
17220 WEND
17230 WHILE cc-1=qpos
17240 PEN 3:PRINT
17250 PRINT "**************"
17260 PRINT "*            *"
17270 PRINT "* CORRECTO! *"
17280 PRINT "*            *"
17290 PRINT "**************"
17300 FOR i=800 TO 100 STEP -100
17310 SOUND 1,i,10
17320 NEXT i
17330 PEN 1
17340 correctas=correctas+1
17350 cc=0:WEND
17360 WHILE cc-1<>qpos AND cc>0
17370 PRINT:PRINT "Lo siento,se ha equiv
ocado. La respues-"
17380 PRINT "ta correcta era ";MID$(list
a$(qprin,0),2);"."
17390 cc=0:WEND
17400 total=total+1
17410 PRINT:INPUT "Mas preguntas (s/n)";
q$
17420 WEND
17430 RETURN
```

Comentarios
Líneas 17040-17100: Ya hemos hecho observar que Multirrespuesta es capaz de establecer dos niveles de tests. Estas líneas permiten que el usuario especifique si las respuestas posibles se deben extraer de todo el fichero o solamente de entre las del mismo tipo que la respuesta correcta.
Líneas 17150-17190: Se presentan en pantalla la pregunta y las cinco posibles respuestas (que serán seleccionadas por el módulo siguiente). Las posiciones de las cinco posibles respuestas están guardadas en la matriz QU, y la posición de la respuesta correcta dentro de QU está registrada por la variable QPOS.
Líneas 17230-17350: Si la respuesta escogida por el usuario, según está representada por la variable CC, se corresponde con la posición de la respuesta correcta (QPOS), entonces se visualiza la palabra «CORRECTO» y suena una especie de trino. Se incrementa en 1 a la variable CORRECTAS, que registra el número de respuestas correctas. Si se da una respuesta errónea, se le informa al usuario de tal circunstancia y se le dice cuál era la correcta.
Línea 17400: La variable TOTAL es incrementada en 1 (esta variable registra el número total de preguntas efectuadas).

Módulo 4.4.14: Selección de las preguntas aleatorias

Habiéndonos proporcionado la capacidad de visualizar las preguntas y las respuestas, pasaremos a la tarea considerablemente más compleja de *seleccionar* dichas preguntas y respuestas. Antes de llegar a los comentarios pormenorizados, echaremos un vistazo general al método requerido.

Lo que queremos es seleccionar una pregunta y su respuesta correcta correspondiente, y rellenar entonces la matriz QU con 5 números que representen la posición dentro del fichero de cinco respuestas potenciales (incluyendo la correcta). En primer lugar se escoge aleatoriamente, de entre todo el fichero, la pregunta y respuesta principales, y el número de dicha pregunta/respuesta dentro de la matriz principal se coloca en una posición aleatoria dentro de la matriz QU.

Habiendo colocado la pregunta principal en QU, hay que encontrar ahora cuatro respuestas alternativas. Dependiendo de que el usuario desee la versión fácil o difícil del test, las cuatro respuestas alternativas se seleccionarán de entre todo el fichero o de la sección que contiene respuestas cuyo tipo es el mismo que el de la respuesta correcta. Las cuatro respuestas se escogen aleatoriamente de la sección apropiada del fichero, haciéndose comprobaciones de que no se

incluye dos veces una misma respuesta y de que no se incluye ninguna respuesta que parezca ser idéntica a la respuesta correcta.

Módulo 4.4.14: Líneas 18000 - 18210

```
18000 REM*******************************
18010 REM Seleccion aleatoria
18020 REM*******************************
18030 qprin=INT(RND*(pr))
18040 ctipo=VAL(LEFT$(lista$(qprin,0),1)
)
18050 r1=ntipo(1,ctipo)
18060 r2=ntipo(0,ctipo)
18070 IF r2<5 OR rq=2 THEN r1=0:r2=pr
18080 qpos=INT(RND*5)
18090 FOR i=0 TO 4
18100 xx=qprin
18110 dup=1:WHILE dup=1 AND i<>qpos
18120 dup=0
18130 xx=r1+INT(RND*r2)
18140 IF MID$(lista$(xx,0),2)=MID$(lista
$(qprin,0),2) THEN dup=1
18150 FOR j=0 TO i-1
18160 IF MID$(lista$(xx,0),2)=MID$(lista
$(qu(j),0),2) THEN dup=1
18170 NEXT j
18180 WEND
18190 qu(i)=xx
18200 NEXT i
18210 RETURN
```

Comentarios

Líneas 18030-18040: Se seleccionan la pregunta y respuesta principales, y su posición se almacena en QPRIN. El tipo de la respuesta es registrado por la variable CTIPO.

Líneas 18050-18070: Las dos variables R1 y R2 registran el comienzo y el final de la parte del fichero de la que se van a extraer las preguntas. Si el usuario ha especificado el tipo de test más difícil, entonces R1 y R2 se fijan de forma que apunten al principio y al final del grupo de cuestiones que son del mismo tipo que la pregunta principal. Si sucediera que hay menos de cinco respuestas en ese grupo (de forma que sería imposible escoger cinco respuestas diferentes), o si el usuario ha especificado la forma más fácil del test, entonces R1 y R2 se fijan al comienzo y al final del fichero. Si especifica la forma más difícil del test, y observa que el programa no actúa en conse-

cuencia, la razón más probable es que no hay suficientes respuestas del mismo tipo que la pregunta principal.
Línea 18080: QPOS representa la posición de la respuesta correcta dentro de la matriz de cinco respuestas posibles.
Líneas 18090-18200: Este bucle escoge las cuatro respuestas alternativas de la parte de fichero indicada por R1 y R2, y almacena provisionalmente cada una de ellas en la variable XX. Dentro del bucle se hacen comprobaciones que consisten en la comparación de la nueva respuesta con la respuesta correcta (que puede estar en cualquier sitio dentro de QU) y con las respuestas que previamente ya hayan sido colocadas en QU. Observe que no es suficiente la simple comprobación de que la misma respuesta del fichero principal no está duplicada. Dos preguntas de partes diferentes del fichero principal pueden muy bien tener idénticas respuestas. Al final del bucle, QU contiene las posiciones de cinco respuestas diferentes del fichero principal.

Comprobación

La única forma efectiva de comprobar estos módulos es introducir un número suficiente de datos como para permitir que el test se pueda generar. La mejor comprobación breve sería registrar primero seis tipos de preguntas, titulados TIPO 1, TIPO 2...TIPO 6. Después introducir una serie de preguntas en la forma P1, P2... P10, con respuesta de la forma R1, R2... R10. El tipo de las cinco primeras preguntas debería ser el TIPO 1, y el resto deberían ser de los tipos TIPO 2...TIPO 6 (cada una un tipo distinto). Esto proporciona un conjunto de preguntas capaz de generar la forma más difícil de test, y otros cinco que sólo contienen una pregunta cada uno de ellos.

Llame al generador de preguntas aleatorias y especifique la forma más difícil de test. Debería encontrar que pueda continuar respondiendo preguntas siendo informado correctamente de si sus respuestas son correctas o erróneas. Cuando se escoge una pregunta de las cinco primeras, las cinco respuestas deberían estar siempre en el intervalo 1 a 5; mientras que cuando se escogen otras preguntas, debería observar que las respuestas se extraen de la totalidad del fichero.

Módulo 4.4.15: Cálculo de la puntuación

El toque final que daremos al programa es permitirle que calcule una puntuación significativa de las respuestas dadas al test. Esto no es tan sencillo como parece, ya que no es solamente una cuestión de calcular el porcentaje de respuestas acertadas respecto a las totales.

Si el usuario se limita a contestar siempre la primera respuesta en cada test, en promedio acertaría una respuesta de cada cinco preguntas; por lo que una puntuación del 20 % puede muy bien indicar que el usuario no tiene la menor idea de cuáles son las respuestas correctas. La solución adoptada es restar un quinto del total de preguntas formuladas (el número que cabría esperar por puro azar) de las respuestas acertadas, y expresar ese número en forma de porcentaje respecto a cuatro quintos del número total de preguntas.

Módulo 4.4.15: Líneas 19000 - 19110

```
19000 REM*******************************
19010 REM Puntuacion
19020 REM*******************************
19030 CLS
19040 PRINT "Puntuacion:":PRINT
19050 IF total=0 THEN x$=" **TODAVIA NO
HAY NINGUNA PUNTUACION**  ":GOSUB 25000:
RETURN
19060 PRINT "Total de respuestas:";total
19070 PRINT "Respuestas correctas:";corr
ectas
19080 PRINT:PRINT "Puntuacion:";INT((cor
rectas-total/5)/(total*0.8)*100+0.5);"%"
19090 PRINT:INPUT  "Desea poner la puntu
acion a cero (s/n) ";q$
19100 IF LOWER$(q$)="s" THEN total=0:cor
rectas=0
19110 RETURN
```

Comprobación

Realice de nuevo la comprobación del módulo anterior. Cuando haya contestado a unas cuantas preguntas, vuelva al menú principal y llame al módulo de puntuación. Debería encontrar que la puntuación dada tiene cierto sentido, aún cuando no sea fácil correlacionarla exactamente con el número de respuesta que haya dado. Debería tener también la opción de poner la puntuación a cero y poder comenzar un nuevo test desde el principio.

Si el funcionamiento de esta comprobación es satisfactorio, el programa está listo para ser usado.

5. Cuestiones monetarias

En este último capítulo desviaremos nuestra atención hacia un aspecto importante que hasta ahora hemos pasado por alto: el 464 y el dinero. Se trata de un tema que, desde un punto de vista realista, no puede ser ignorado, dado que los microordenadores se ocupan tan magníficamente bien de los temas financieros. Las cantidades implicadas raramente son inmensas (y si lo son es muy improbable que se vayan a tratar con un micro económico), y los cálculos relacionados son generalmente sencillos: una cuestión de sumar y restar, según el dinero entra o se va.

Sin embargo, la ventaja real del microordenador no es simplemente que pueda hacer cálculos monetarios, eso también lo puede hacer el cerebro humano. El microordenador tiene ventaja en cuanto a su capacidad de almacenar información, para recuperarla rápidamente y presentarla de tal forma que pueda ser entendida inmediatamente. Los dos ejemplos de este capítulo ilustran esta capacidad, uno de ellos permitiendo al usuario seguir de cerca una cuenta bancaria, y el otro permitiendo llevar un conjunto sencillo de cuentas.

Los dos programas que se incluyen en este capítulo son:

BANQUERO: Que guarda un registro permanente de los ingresos y pagos que se realizan en una cuenta bancaria.

CONTABLE: Que compila una serie de cuentas en formato tradicional.

PROGRAMA 5.1: Banquero

Función del programa

El propósito de este programa es permitir que el usuario pueda llevar un registro continuo y actualizado de una única cuenta bancaria: denominación de los pagos, fecha y cantidad, e incluyendo la capacidad de especificar no solamente pagos únicos sino pagos o ingresos periódicos, independientemente de la irregularidad del período. El pro-

grama está diseñado para ocuparse de una cuenta durante un período de un año civil.

```
                    ENERO
-------------------------------------
DIA & CONCEPTO      CANTIDAD      SALDO
-------------------------------------
   Saldo anterior                 0.00
 2 ALIMENTACION     9590.50-    9590.50-
 4 GAS              4924.75-   14515.25-
15 NOMINA         119256.00   104740.75
17 TELEFONO         9897.25-   94843.50
19 HIPOTECA        45100.00-   49743.50
21 GARAJE           3250.00-   46493.50
26 LUZ              7257.60-   39235.90
```

Fig. 5.1 Vaciado de pantalla del programa Banquero.

Módulo 5.1.1: Inicialización

Un módulo estándar de inicialización.

Módulo 5.1.1: Líneas 10000 - 10160

```
10000 REM*******************************
10010 REM Inicializacion
10020 REM*******************************
10030 MODE 1
10040 WHILE in=0
10050 in=1:rc$=CHR$(13)
10060 DIM a$(99,1),a(99,1):a(0,1)=999
10070 RESTORE
10080 DIM mes$(11)
10090 FOR i=0 TO 11
10100 READ mes$(i)
10110 NEXT i
10120 DATA Enero,Febrero,Marzo,Abril,May
o,Junio
10130 DATA Julio,Agosto,Septiembre,Octub
re,Noviembre,Diciembre
10140 WEND
10150 INK 0,24:INK 1,3:INK 2,12
10160 WINDOW #1,1,40,20,25
```

Comentarios

Línea 10060: La matriz A% se utilizará para almacenar los nombres de los pagos y una cadena especial, que se explicará más tarde, ca-

dena que registra el mes en particular en el que se ha hecho el pago. La matriz numérica A almacenará la cantidad correspondiente a cada pago y el día del mes en el que se hace. El ajuste de A(0,1) a 999, un día imposible de mes, se utiliza por la ulterior rutina de clasificación para detectar el fin del fichero.
Líneas 10080-10130: Este bucle lee e introduce los nombres de los meses del año en la matriz MES$.

Módulo 5.1.2: El menú del programa

Un módulo de menú estándar, con el servicio adicional de que el usuario es informado en caso de que le pida al programa que saque resultados antes de que se haya introducido ningún dato.

Módulo 5.1.2: Líneas 11000 - 11210

```
11000 REM*******************************
11010 REM Menu
11020 REM*******************************
11030 CLS:CLS #1:PEN 2:PRINT TAB(16);"BA
NQUERO";TAB(16);"========":PEN 1:WINDOW
1,40,4,25
11040 z=0:WHILE z<>6:CLS
11050 PRINT "Ordenes disponibles:":PRINT
11060 PRINT "1) Nuevas partidas."
11070 PRINT "2) Examinar/Suprimir partid
as."
11080 PRINT "3) Presentar estado de cuen
tas."
11090 PRINT "4) Grabar fichero."
11100 PRINT "5) Cargar fichero
11110 PRINT "6) Terminar."
11120 PRINT:INPUT "Cual elige: ",z
11130 WHILE pa=0 AND (z=2 OR z=3 OR z=4)
11140 PRINT:PRINT:PRINT " *AUN NO SE HA
INTRODUCIDO NINGUN DATO* ":SOUND 1,1000,
100:FOR i=1 TO 1500:NEXT i
11150 z=0
11160 WEND
11170 ON z GOSUB 12000,13000,14000,15000
,16000
11180 WEND
11190 CLS
11200 LOCATE 6,11:PRINT "ESTA OFICINA AC
ABA DE CERRAR"
11210 END
```

Módulo 5.1.3: Introducción de nuevas partidas

Se trata de un módulo de entrada más complejo que los que hemos estado utilizando hasta ahora, por la sencilla razón de que las entradas en sí son más complejas. Por cada partida registrada se necesitan conocer cinco datos: si el asiento es haber o debe (dinero recibido o dinero a pagar), el nombre de la partida, la cantidad, los meses en que se efectúa el pago o ingreso y el día del mes en que se hace.

Módulo 5.1.3: Líneas 12000 - 12400

```
12000 REM*******************************
12010 REM Introduccion de nuevas parti-
      das
12020 REM*******************************
12030 CLS
12040 PRINT "Nuevas partidas:":PRINT
12050 PRINT "1) Haber":PRINT "2) Debe"
12060 PRINT:INPUT "Cual desea: ",hd:hd=h
d-1
12070 q$="":WHILE q$="" OR LEN(q$)>14:PR
INT:PRINT "Nombre del pago (14 caractere
s maximo):":INPUT;q$:WEND
12080 PRINT:INPUT "Cantidad:",q
12090 ne=1
12100 WHILE ne
12110 LOCATE #1,1,1:INPUT #1,"Meses (P.e
j.:01040710):",r$:PRINT #1
12120 ne=0:FOR i=1 TO LEN(r$) STEP 2
12130 m=VAL(MID$(r$,i,2))-1
12140 IF m<0 OR m>11 THEN PRINT #1," ***
**   DATO DE MES INCORRECTO   ***** ":SO
UND 1,1000,100:ne=1:i=LEN(r$):FOR j=1 TO
 1500:NEXT j
:PRINT #1,STRING$(240," ")
12150 IF ne=0 THEN PRINT #1,mes$(m);"/";
12160 NEXT i
12170 WEND
12180 INPUT "Dia del pago:",s
12190 PRINT:INPUT "Es todo correcto (s/n
)";t$
12200 IF LOWER$(t$)<>"s" THEN RETURN
12210 pa=pa+1
12220 j=pa-1
12230 WHILE s<a(ABS(j),1) AND j>=0
12240 FOR k=0 TO 1
```

```
12250 a$(j+1,k)=a$(j,k)
12260 a(j+1,k)=a(j,k)
12270 NEXT k
12280 j=j-1
12290 WEND
12300 j=j+1
12310 a$(j,1)="000000000000"
12320 FOR i=1 TO LEN(r$) STEP 2
12330 m=VAL(MID$(r$,i,2))
12340 a$(j,1)=LEFT$(a$(j,1),m-1)+"1"+RIG
HT$(a$(j,1),12-m)
12350 NEXT i
12360 a$(j,0)=q$
12370 a(j,0)=q
12380 a(j,1)=s
12390 IF hd=1 THEN a(j,0)=-a(j,0)
12400 RETURN
```

Comentarios

Líneas 12040-12060: Evidentemente el programa necesita saber si la partida es algo a pagar o a recibir, debe o haber. Esto se registra en la variable HD (Haber/Debe).

Líneas 12090-12170: Los meses en que se efectúa el pago se introducen en forma de una cadena de números de dos dígitos sin separación entre ellos. Así, si se fuera a necesitar un pago trimestral en febrero, mayo, agosto y noviembre, la entrada sería «02050811», que representa a los meses 2, 5, 8 y 11. El bucle FOR que comienza en la línea 12120 explora la cadena introducida para asegurar que proporciona una serie de valores de mes lógicos, e informa al usuario si se ha cometido algún error. Como última comprobación, el bucle presenta en pantalla los nombres de los meses especificados, según estaban registrados en MES$, de forma que el usuario puede determinar que no son solamente lógicos sino que, de hecho, son los meses que se pretendían. La rutina que comienza en la línea 12090 se repite en caso de que se haya introducido un mes no válido, ya que el bucle WHILE depende del valor de NE (Número de Error) que se activa (a 1) si se hace una entrada incorrecta. Observe el uso de una ventana para la entrada relacionada con los meses, ventana que permite que se puedan borrar las líneas particulares implicadas sin perder de vista los otros datos que ya se hayan introducido.

Línea 12210: En este punto, la información introducida ha sido comprobada y confirmada por el usuario, por lo que la variable que registra el número de partidas del fichero se incrementa en 1.

Líneas 12230-12300: El propósito de este bucle es colocar la nueva partida, ordenada según el día, dentro del fichero. En vez de hacer

copias duplicadas de los pagos que se realizan en más de un mes, el sistema adoptado es almacenar todas las entradas una única vez, ordenadas según *días*. Cuando se pide el estado de cuentas de un mes en particular, una parte posterior del programa recorrerá todas las entradas, comprobando cada una de ellas para ver si pertenecen o son anteriores al mes especificado y necesitan, por tanto, ser tenidas en cuenta para determinar el saldo.

Cuando se introduce la nueva partida, el bucle que va de la línea 12230 a la 12290 comienza por el valor de día más alto (que es el 999. ficticio insertado en el módulo de inicialización) y se abre camino hacia abajo hasta que encuentra un pago con un valor de día *menor* que S (el valor de día de la partida recién introducida por el usuario). En tanto *no* encuentre la posición correcta, mueve un lugar hacia arriba a la partida que acaba de examinar. En otras palabras, conforme va explorando hacia abajo en el fichero, arrastra una línea libre con él. Cuando encuentra la posición correcta, la línea libre ya está colocada también en su posición correcta. Como última acción, el valor de J (que registra la posición de la primera entrada encontrada con un día de pago inferior a S) se incrementa en 1 para que apunte hacia la línea libre.

Líneas 12310-12350: La tarea siguiente es transformar la lista de meses introducida en un formato que pueda ser fácilmente explorado por partes posteriores del programa. El simple recurso adoptado es utilizar una cadena de 12 ceros (registrando meses en que *no* se va a hacer el pago), y utilizar entonces un bucle para cambiar un cero a uno en aquellos meses en que el pago *sí* se va a hacer. Así, en el caso de nuestro ejemplo trimestral de antes, la cadena final quedaría como «010010010010».

Líneas 12360-12390: La nueva información se coloca en las matrices principales. Si la variable HD registra que la partida es un débito, la cantidad es multiplicada por menos uno, haciéndola negativa.

Comprobación

Lance el programa y llame a la opción 1 desde el menú. Introduzca una nueva partida con los siguientes datos:

Debe (contestar opción 2 a la pregunta)
Nombre: COMPROBACION
Cantidad: 100
Meses: 02050811
Día: 15

Tras una pausa se debería volver al menú principal. Termine el programa utilizando la opción 6 del menú. Ahora teclee:

PRINT A$(0,0),A$(0,1),A(0,0),A(0,1)

El resultado debería ser:

COMPROBACION 010010010010 –100 15

Módulo 5.1.4: Presentación del estado de cuentas

Aunque quedan más módulos por venir, la tarea final de la parte principal del programa es coger las partidas que se han introducido utilizando el módulo anterior, y compilarlas en un estado de cuentas para cualquier mes especificado del año. El estado de cuentas incluirá un cálculo del suma y sigue de los meses anteriores, y presentará al completo todas las partidas del mes y el saldo continuo generado tras cada partida.

Módulo 5.1.4: Líneas 14000 - 14290

```
14000 REM*******************************
14010 REM Compilar estado de cuentas
14020 REM*******************************
14030 sum=0
14040 CLS:PRINT "Estado de cuentas:":PRI
NT
14050 q=0:WHILE q<1 OR q>12
14060 PRINT:INPUT "De que mes (1 a 12):"
,q
14070 WEND
14080 FOR j=1 TO q-1
14090 FOR i=0 TO pa-1
14100 IF MID$(a$(i,1),j,1)="1" THEN sum=
sum+a(i,0)
14110 NEXT i
14120 NEXT j
14130 PRINT:INPUT "Desea sacar el estado
 de cuentas por la impresora (s/n)";p$:c
a=0:IF LOWER$(p$)="s" THEN ca=8
14140 CLS:PRINT #ca,TAB(20-LEN(mes$(q-1)
)/2);UPPER$(mes$(q-1)):PRINT #ca,STRING$
(39,"-")
14150 PRINT #ca,"DIA & CONCEPTO";TAB(20)
;"CANTIDAD      SALDO"
14160 PRINT #ca,STRING$(39,"-")
14170 PRINT #ca,"   Saldo anterior";:PEN
 2-SGN(sum+0.001):PRINT #ca,TAB (30);USI
```

```
NG "######.##-";sum
14180 FOR i=0 TO pa-1
14190 d=0:WHILE MID$(a$(i,1),q,1)="1" AN
D d=0:d=1
14200 PEN 1:PRINT #ca,USING "## &";a(i,1
);a$(i,0);
14210 PEN 2-SGN(a(i,0+0.001)):PRINT #ca,
TAB(19);USING "######.##-";a(i,0);
14220 sum=sum+a(i,0)
14230 PEN 2-SGN(sum+0.001):PRINT #ca,TAB
(30);USING "######.##-";sum
14240 WEND
14250 NEXT i:PEN 1
14260 IF ca=8 THEN RETURN
14270 PRINT:PRINT "Pulse cualquier tecla
 para continuar.  "
14280 WHILE INKEY$="":WEND
14290 RETURN
```

Comentarios

Línea 14030: La variable SUM se utilizará para guardar el saldo de la cuenta (tanto el suma y sigue de meses anteriores, como el saldo tras cada partida).

Líneas 14080-14120: Supuesto que el estado de cuentas no es para el primer mes, en cuyo caso no hay suma y sigue de meses anteriores, estos bucles exploran la totalidad de las partidas listadas alguna vez en cada mes anterior al mes del que se pide el estado de cuentas. De esta forma, se examina cada partida para ver si se hace en cualquiera de los meses anteriores, en cuyo caso la cantidad apropiada se suma al total en SUM. Al final de los dos bucles, SUM contiene el total de todos los cambios del saldo desde el comienzo del año. El mantenimiento de un saldo total, que incluya cualquier cantidad que estuviera en la cuenta al comienzo del año, se puede conseguir fácilmente introduciendo el saldo correspondiente al final del año anterior como haber en el primer día de enero.

Línea 14170: Una cosa a tener en cuenta, en este punto, para la presentación de SUM, es que si la cantidad es negativa el color de impresión se cambia a rojo. En las líneas que siguen encontrará frecuentemente las mismas técnicas. Observe también la utilización de la orden PRINT USING, que nos permite imponer un formato estándar al número que se va a presentar, y que asegura que el estado de cuentas se presentará de forma elegante, con todos los puntos decimales alineados.

Líneas 14180-14250: Este bucle explora la lista completa de partidas, mientras que el bucle de las líneas 14190-14120 selecciona sola-

mente aquellas que tienen un «1» en la posición pertinente de la cadena que registra los meses en que se realiza esa partida. Cuando una partida se efectúa en el mes especificado en el estado de cuentas, el bucle presenta el día (A(I,1)), el nombre (A$(I,0)), la cantidad (A(I,0)), y por último el saldo resultante, obtenido sumando la cantidad al total anterior de SUM. Aparte del día, las cantidades correspondientes a un debe se presentan en rojo. La pantalla mantiene una presentación ordenada en columnas, a pesar de que las cifras pueden variar en longitud, gracias al uso de TAB (que hace que las partidas comiencen en una posición estándar de la pantalla) y de PRINT USING.

Comprobación

Introduzca los datos de comprobación que utilizó en el último módulo. Cuando aparezca el menú principal, especifique la opción 3 para presentar el estado de cuentas.

Comience por el mes 1 y, cuando esté satisfecho, vuelva al menú principal y pida el estado de cuentas del mes siguiente. Debería encontrar que la mayoría de los estados están en blanco, ya que no hay ninguna partida (ingreso o pago) en la mayoría de los meses. Sin embargo, los meses de febrero, mayo, agosto y noviembre tienen una partida que debería quedar claramente visualizada.

Módulo 5.1.5: Ficheros de datos

Un módulo estándar de fichero de datos.

Módulo 5.1.5: Líneas 15000 - 16140

```
15000 REM*******************************
15010 REM Grabar en cinta
15020 REM*******************************
15030 CLS:q$="":WHILE LOWER$(q$)<>"s"
15040 INPUT "Nombre del fichero que se v
a a grabar:";fi$
15050 PRINT:PRINT "El fichero que se va
a grabar es ";fi$
15060 PRINT:INPUT "Es correcto (s/n)";q$
15070 WEND
15080 OPENOUT fi$
15090 PRINT #9,pa
15100 FOR i=0 TO pa-1
15110 PRINT #9,a$(i,0);rc$;a$(i,1);rc$;a
(i,0);rc$;a(i,1)
15120 NEXT i
```

```
15130 CLOSEOUT
15140 RETURN
16000 REM*******************************
16010 REM Cargar desde cinta
16020 REM*******************************
16030 CLS:q$="":WHILE LOWER$(q$)<>"s"
16040 INPUT "Nombre del fichero que se v
a a cargar:";fi$
16050 PRINT:PRINT "El fichero que se va
a cargar es ";fi$
16060 PRINT:INPUT "Es correcto (s/n)";q$
16070 WEND
16080 OPENIN fi$
16090 INPUT #9,pa
16100 FOR i=0 TO pa-1
16110 INPUT #9,a$(i,0),a$(i,1),a(i,0),a(
i,1)
16120 NEXT i
16130 CLOSEIN
16140 RETURN
```

Módulo 5.1.6: Cambio y supresión de partidas

Igual que en Nnúmero, se trata de un módulo muy sencillo para permitir cambios por parte del usuario, trabajando esta vez sobre la base de un bucle FOR que explora las partidas una a una.

Módulo 5.1.6: Líneas 13000 - 13250

```
13000 REM*******************************
13010 REM Examinar/Suprimir partidas
13020 REM*******************************
13030 FOR i=0 TO pa-1
13040 CLS
13050 PRINT "Partida:";a$(i,0)
13060 PRINT "Cantidad:";a(i,0)
13070 PRINT "Meses:";
13080 FOR j=1 TO 12
13090 IF MID$(a$(i,1),j,1)="1" THEN PRIN
T mes$(j-1);"/";
13100 NEXT j:PRINT
13110 PRINT "Dia del pago:";a(i,1)
13120 PRINT:PRINT "Ordenes disponibles:"
:PRINT
13130 PEN 2:PRINT "ENTER";:PEN 1:PRINT "
 para ver la partida siguiente."
```

```
13140 PRINT "'\' para volver al menu."
13150 PRINT "'\S' para suprimir la parti
da."
13160 q$="":PRINT:INPUT "Cual desea:",q$
13170 WHILE UPPER$(q$)="\S"
13180 FOR j=i TO pa-1
13190 FOR k=0 TO 1
13200 a$(j,k)=a$(j+1,k):a(j,k)=a(j+1,k)
13210 NEXT k,j
13220 pa=pa-1:q$="\"
13230 WEND
13240 IF q$="" THEN NEXT i
13250 RETURN
```

Comprobación

Llame a esta opción una vez que tenga algunos datos introducidos y grabados en cinta. Debería poder ir recorriendo las distintas partidas, borrándolas o dejándolas inalteradas a voluntad. Si estas funciones están disponibles, entonces el programa funciona al completo.

PROGRAMA 5.2: Contable

Función del programa

El segundo programa de este último capítulo es más complejo que Banquero. Su función es la de llevar los dos lados de un conjunto sencillo de cuentas, presentándolas en formato tradicional, con algunas partidas en solitario y con otras divididas claramente en grupos que representan diferentes tipos de gastos. Tal y como se presenta, el programa está proyectado para arreglárselas con cantidades de hasta 999.999,99 Pt. Las cifras mayores que esta cantidad se calculan con igual precisión, pero echan a perder el formato de presentación de las cuentas. Si va a utilizar conceptos o totales de 1.000.000 o superiores, modifique simplemente las órdenes PRINT USING y rehaga el espaciado de la presentación.

Módulo 5.2.1: Inicialización

Un módulo estándar de inicialización.

Módulo 5.2.1: Líneas 10000 - 10080

```
10000 REM*******************************
10010 REM Inicializacion
10020 REM*******************************
10030 CLS
10040 WHILE in=0
10050 in=1:rc$=CHR$(13):
10060 DIM a$(1,99),a(1,99)
10070 WEND
10080 MODE 1
```

```
                        DEBE

GASTOS COCHE
  NEUMATICOS          11355.00
  GASOLINA            13450.00
  REPARACIONES        20624.20
                      ---------
                                   45429.20
AMSTRAD CPC 464                    95900.00

GASTOS CASA
  CONTRIBUCION        26650.00
  LUZ                 11850.40
  GAS                  9647.00
                      ---------
                                   48147.40
VACACIONES                         55600.00
                                   ---------
TOTAL:                            245076.60
```

Fig. 5.2 Vaciado de pantalla del programa Contable.

Comentarios
Línea 10060: Los dos lados de las cuentas (haber y debe), y los nombres asociados a cada concepto, son almacenados en los dos lados de las matrices A y A$. Se pueden almacenar hasta 100 conceptos en ambos lados, aunque evidentemente dicho número se puede aumentar sin ninguna dificultad.

Módulo 5.2.2: El menú principal

Un módulo estándar de menú, con protección al acceso de ciertas funciones mientras que no se haya introducido ningún dato.

Módulo 5.2.2: Líneas 11000 - 11220

```
11000 REM*****************************
11010 REM Menu
11020 REM*****************************
11030 CLS:PEN 2:PRINT TAB(16);"CONTABLE"
;TAB(16);"========":PEN 1:WINDOW 1,40,4,
25
11040 z=0:WHILE z<>6:CLS
11050 PRINT "Ordenes disponibles:":PRINT
11060 PRINT "1) Nuevos titulos."
11070 PRINT "2) Cambiar/Suprimir partida
s."
11080 PRINT "3) Presentar cuentas."
11090 PRINT "4) Grabar fichero."
11100 PRINT "5) Cargar fichero."
11110 PRINT "6) Terminar."
11120 PRINT:INPUT "Cual desea:",z
11130 IF z<4 AND z>0 THEN GOSUB 12000
11140 WHILE c(hd)=0 AND (z=2 OR z=3 OR z
=4)
11150 PRINT:PRINT:PRINT " *AUN NO SE HA
INTRODUCIDO NINGUN DATO* ":SOUND 1,1000,
100:FOR i=1 TO 1500:NEXT i
11160 z=0
11170 WEND
11180 ON z GOSUB 13000,16000,18000,19000
,20000
11190 WEND
11200 CLS
11210 LOCATE 8,11:PRINT "EL PROGRAMA HA
TERMINADO"
11220 END
```

Módulo 5.2.3: ¿Haber o debe?

Al contrario que en Banquero, hay varias partes del programa que necesitan saber si se está especificando un haber o un debe, es por ello que la rutina que pide esta información se incluye en un módulo independiente.

Módulo 5.2.3: Líneas 12000 - 12110

```
12000 REM*****************************
12010 REM Haber o debe?
12020 REM*****************************
```

```
12030 hd=-1:WHILE hd<>0 AND hd<>1
12040 CLS
12050 PRINT "Haber o debe:":PRINT
12060 PRINT "1) Haber.":PRINT "2) Debe."
12070 PRINT:INPUT "Que desea: ",hd:hd=hd
-1
12080 hd$="HABER"
12090 IF hd=1 THEN hd$="DEBE"
12100 WEND
12110 RETURN
```

Módulo 5.2.4: Tipo de partida

Este módulo es trivial en sí mismo, pero da una pista de por qué este programa está obligado a ser más largo que otro del tipo de Banquero. El propósito del módulo es permitirle al usuario que especifique a cuál de los tres tipos pertenece la partida que va a introducir. Los tres tipos son:

1) Una partida única: todo lo que se necesita en este caso es el nombre de la partida y la cantidad. Cuando finalmente la cuenta sea presentada, las partidas individuales tendrán sus nombres impresos en la parte izquierda, y la cantidad asociada en la columna principal de cifras de la derecha.
2) Un encabezamiento: este tipo es el que permite que se especifiquen grupos de partidas dentro de la cuenta total. Si, por ejemplo, estuviera utilizando el programa para llevar la contabilidad doméstica, podría poner «COCHE» como encabezamiento de un grupo que incluya partidas como neumáticos, gasolina, reparaciones, etc. En la cuenta final, el nombre del encabezamiento se presentará en el lado izquierdo, pero no se presentará ninguna cantidad frente a dicho encabezamiento.
3) Subtítulos: como se ilustró en el punto 2, cada encabezamiento puede tener una lista de partidas asociadas a él, partidas que forman parte de un grupo independiente. En las cuentas, los nombres de los subtítulos se presentarán bajo su encabezamiento pertinente (sangrados en unos cuantos espacios), y con la cantidad asociada a cada uno de ellos presentada a la izquierda de la columna principal de cantidades.

Módulo 5.2.4: Líneas 13000 - 13120

```
13000 REM*******************************
13010 REM Introducir titulos
13020 REM*******************************
13030 CLS
13040 PRINT "Nuevas partidas:":PRINT
13050 PRINT hd$:PRINT
13060 PRINT "Que clase de partida desea?
:":PRINT
13070 PRINT "1) Una partida unica."
13080 PRINT "2) Un encabezamiento."
13090 PRINT "3) Un subtitulo."
13100 PRINT:INPUT "(0 para volver al men
u)";tipo
13110 ON tipo GOSUB 14000,14000,15000
13120 RETURN
```

Comprobación

Habiendo introducido todas las partes del programa que no ejecutan cálculos, probablemente lo mejor es que lance el programa y compruebe rápidamente el menú. Si especifica que desea introducir una nueva partida, debería ser preguntado sobre si se trata de un haber o de un debe, y entonces se le pedirá que especifique el tipo (y hasta aquí es hasta donde puede llegar). La única otra opción del menú que tendrá algún efecto, es la opción 5 para terminar el programa. El propio menú debería impedirle el acceso a las funciones de alteración de datos o de presentación de la cuenta, ya que aún no se ha introducido ningún dato.

Módulo 5.2.5: Introducción de partidas únicas y de encabezamientos

Dos módulos independientes se encargan de la introducción de subtítulos por una parte, y de partidas únicas y encabezamientos por la otra. Es importante, para entender partes posteriores del programa, que intente seguir la forma en que se almacenan las partidas y los caracteres indicadores especiales que registran el tipo de partida.

Módulo 5.2.5: Líneas 14000 - 14120

```
14000 REM*******************************
14010 REM Partida unica o encabezamien-
          to
14020 REM*******************************
```

```
14030 q=0:q$="":r$=""
14040 PRINT:INPUT "Nombre de la partida
o el encabezamiento:",q$
14050 IF tipo=1 THEN PRINT:INPUT "Cantid
ad:",q
14060 PRINT:INPUT "Es correcto (s/n)";r$
14070 IF LOWER$(r$)<>"s" THEN PRINT:PRIN
T "     *****  NO SE HA REGISTRADO *****"
:SOUND 1,1000,100:FOR i=1 TO 1500:NEXT i
:RETURN
14080 IF tipo=1 THEN q$="%"+q$ ELSE q$="
*"+q$
14090 a$(hd,c(hd))=q$
14100 a(hd,c(hd))=q
14110 c(hd)=c(hd)+1
14120 RETURN
```

Comentarios

Línea 14050: Como se mencionó en la introducción del módulo anterior, los encabezamientos no tienen cantidades asociadas, por lo que esta línea acepta cifras únicamente cuando se trata de partidas únicas.

Línea 14080: No existen áreas de almacenamiento independientes para los diferentes tipos de partidas (aparte de los lados debe y haber de las matrices). Las partes posteriores del programa, determinarán el tipo de partida examinando un carácter indicador especial añadido al principio del nombre de la partida. Este indicador será «%» para una partida única y «*» para un encabezamiento.

Líneas 14090-14120: Ya nos hemos encontrado antes con la variable HD, que registra si una partida es haber o debe. Aquí HD se utiliza para decidir en qué lado de las matrices A y A$ se colocará la nueva partida. Además, necesitamos llevar un registro del número de partidas existentes en lo lados haber y debe, ya que normalmente serán diferentes. Esto se hace por medio de la matriz C, matriz que no se declaró en el módulo de inicialización ya que tendrá únicamente dos elementos (C(0)) y (C(1)), correspondientes a los lados haber y debe de las matrices principales. Una vez más, se utiliza HD para indicar cuál de los dos elementos de C se va a utilizar. Aplicando esto podemos ver que cuando se hace una referencia a:

```
A   (    HD   ,    C(HD)   )
1        2          3
```

lo que se quiere decir es:

1) Un elemento de la matriz numérica A.

2) En el lado indicado por el valor de HD, es decir: haber o debe.
3) El primer elemento vacío de ese lado, determinado por lo que ya se ha almacenado.

Comprobación

Lance el programa y llame a la opción de nuevas entradas. Especifique que desea introducir un encabezamiento en el lado del haber, introduzca entonces TITULO como nombre de la partida (no se le debería pedir ninguna cantidad). Llame otra vez a la opción de nuevas entradas y especifique una partida única en el lado del haber, con el nombre PRUEBA y la cantidad 100. Ahora haga exactamente lo mismo pero en el lado del debe de las cuentas.

Termine el programa desde el menú, e introduzca en modo directo:

PRINT A(0,0),A(1,0),A$(0,0),A$(1,0)[ENTER]

con lo que debería ver:

0 0 *TITULO *TITULO

Ahora siga el mismo procedimiento para la línea 1 de las matrices, es decir A(0,1), etc. Debería ver:

100 100 %PRUEBA %PRUEBA

Por último, saque el valor de C(0) y C(1): ambos deberían ser iguales a 2.

Módulo 5.2.6: Introducción de un subtítulo

La cuestión de introducir un subtítulo nuevo no es tan sencilla como en el caso de una partida única. Por cada subtítulo nuevo que se introduce, tiene que comprobarse la presencia del encabezamiento pertinente, y la partida debe colocarse en lugar próximo a su encabezamiento, y no simplemente pegada al final de las partidas previamente almacenadas.

Módulo 5.2.6: Líneas 15000 - 15210

```
15000 REM*******************************
15010 REM Subtitulo
15020 REM*******************************
15030 PRINT:INPUT "Encabezamiento:",q$
15040 q$="*"+q$
15050 p1=-1:FOR i=0 TO c(hd)-1
```

```
15060 IF a$(hd,i)=q$ THEN pl=i+1
15070 NEXT i
15080 IF pl=-1 THEN PRINT:PRINT "*NO HAY
 ENCABEZAMIENTOS CON ESE NOMBRE*":SOUND
1,1000,100:FOR i=1 TO 1500:NEXT i:RETURN
15090 PRINT:INPUT "Nombre del subtitulo:
",q$
15100 PRINT:INPUT "Cantidad:",q
15110 PRINT:INPUT "Es todo correcto? (s/
n)";r$
15120 IF LOWER$(r$)<>"s" THEN PRINT:PRIN
T "     ***** NO SE HA REGISTRADO *****":
SOUND 1,1000,100:FOR i=1 TO 1500:NEXT i:
RETURN
15130 q$="$"+q$
15140 FOR i=c(hd)+1 TO pl+1 STEP -1
15150 a$(hd,i)=a$(hd,i-1)
15160 a(hd,i)=a(hd,i-1)
15170 NEXT i
15180 a$(hd,pl)=q$
15190 a(hd,pl)=q
15200 c(hd)=c(hd)+1
15210 RETURN
```

Comentarios

Líneas 15030-15080: Se pide el nombre del encabezamiento pertinente, y se hace una comprobación (entre las partidas que ya hay en el fichero) de que realmente existe; si no existe, se presenta un mensaje de error y el programa retorna al menú.

Líneas 15140-15200: Como se mencionó anteriormente, el sentido de un subtítulo es que debería aparecer en las cuentas formando parte de un grupo presentado bajo el encabezamiento pertinente. Para conseguir esto de una manera sencilla, el método empleado es almacenarlo en el fichero en una posición próxima a su encabezamiento. La posición del primer subtítulo que sigue al encabezamiento ya se ha encontrado por medio del bucle FOR de la línea 15050, así que todo lo que se necesita hacer es correr todas las partidas desde ese punto y colocar el nuevo subtítulo dentro de la matriz y en la posición correcta.

Comprobación

Introduzca las partidas especificadas en la comprobación del módulo 5.2.5, llame otra vez al módulo de nuevas entradas para colocar un subtítulo nuevo en el lado del haber, subtítulo que denominaremos SUBTITULO y con un valor de 200. Haga lo mismo en el lado del debe.

En modo directo, introduzca lo siguiente:

FOR I=0 TO 2:PRINT A(0,I),A(1,I),A$(0,I),A$(1,I):NEXT I[ENTER]

con lo que debería ver:

0	0	*TITULO	*TITULO
200	200	$SUBTITULO	$SUBTITULO
100	100	%PRUEBA	%PRUEBA

Saque los valores de C(0) y C(1): deberían ser iguales a 3.

Módulo 5.2.7: Ficheros de datos

Puesto que los datos para Contable son bastante complejos, posiblemente sea lo más prudente introducir ya el módulo de fichero de datos, eliminando así la necesidad de tener que reintroducir los datos en cada comprobación. Una vez que se haya introducido el módulo, teclee y grabe los datos especificados en la comprobación del módulo anterior.

Módulo 5.2.7: Líneas 19000 - 20160

```
19000 REM*******************************
19010 REM Grabar en cinta
19020 REM*******************************
19030 CLS:q$="":WHILE LOWER$(q$)<>"s"
19040 INPUT "Nombre del fichero que dese
a grabar:     ";fi$
19050 PRINT:PRINT "El fichero que se va
a grabar es ";fi$;"."
19060 PRINT:INPUT "Es correcto (s/n)";q$
19070 WEND
19080 OPENOUT fi$
19090 FOR i=0 TO 1
19100 PRINT #9,c(i)
19110 FOR j=0 TO c(i)-1
19120 PRINT #9,a$(i,j);rc$;a(i,j)
19130 NEXT j
19140 NEXT i
19150 CLOSEOUT
19160 RETURN
20000 REM*******************************
20010 REM Cargar desde la cinta
20020 REM*******************************
20030 CLS:q$="":WHILE LOWER$(q$)<>"s"
```

```
20040 INPUT "Nombre del fichero que dese
a cargar:     ";fi$
20050 PRINT:PRINT "El fichero que se va
a cargar es ";fi$;"."
20060 PRINT:INPUT "Es correcto (s/n)";q$
20070 WEND
20080 OPENIN fi$
20090 FOR i=0 TO 1
20100 INPUT #9,c(i)
20110 FOR j=0 TO c(i)-1
20120 INPUT #9,a$(i,j),a(i,j)
20130 NEXT j
20140 NEXT i
20150 CLOSEIN
20160 RETURN
```

Módulo 5.2.8: Cambios en las partidas

Un módulo sencillo que añade ciertas características para tener en cuenta el hecho de que algunas partidas no van solas sino que forman parte de grupos incluidos bajo un encabezamiento común.

Módulo 5.2.8: Líneas 16000 - 16290

```
16000 REM*******************************
16010 REM Cambios y supresiones
16020 REM*******************************
16030 FOR i=0 TO c(hd)-1
16040 d=0:WHILE d=0:d=1
16050 CLS
16060 PRINT "Cambiar o Suprimir:":PRINT
16070 IF LEFT$(a$(hd,i),1)<>"$" THEN PRI
NT MID$(a$(hd,i),2);
16080 IF LEFT$(a$(hd,i),1)="*" THEN hh$=
MID$(a$(hd,i),2):PRINT
16090 IF LEFT$(a$(hd,i),1)="$" THEN PRIN
T hh$:PRINT "  ";MID$(a$(hd,i),2);
16100 IF a(hd,i)<>0 THEN PRINT TAB(30);U
SING "#######.##";a(hd,i)
16110 PRINT:PRINT "Ordenes disponibles:"
:PRINT
16120 PRINT "1) Ver partida siguiente."
16130 PRINT "2) Cambiar la cantidad."
16140 PRINT "3) Volver al menu."
16150 PRINT "4) Suprimir la partida."
16160 PRINT:PRINT "Cual desea?"
```

```
16170 q$="":WHILE q$=""
16180 q$=INKEY$
16190 WEND
16200 IF q$="4" THEN GOSUB 17000:RETURN
16210 IF q$="3" THEN RETURN
16220 WHILE q$="2" AND LEFT$(a$(hd,i),1)
<>"*"
16230 PRINT:INPUT "Cantidad a incrementa
r (utilice signo   '-' si desea reducir)
:",q
16240 PRINT:INPUT "Es correto? (s/n)";r$
16250 IF LOWER$(r$)="s" THEN a(hd,i)=a(h
d,i)+q:q$=""
16260 WEND
16270 WEND
16280 NEXT i
16290 RETURN
```

Comentarios

Líneas 16070-16090: Si la partida llamada desde el fichero es una partida única, entonces es presentada en pantalla (aunque despojada del carácter indicador que se le añadió al comienzo del nombre). Si la partida es un encabezamiento, no sólo se presenta en pantalla sino que su nombre se almacena en HH$, de forma que pueda visualizarse sobre cualquiera de los subtítulos que le siguen.

Líneas 16220-16260: Además de suprimir partidas, se pueden hacer cambios en sus valores asociados. Esto se consigue añadiendo la cantidad positiva o negativa en que queremos cambiar el valor de la partida (es decir: no se introduce el nuevo valor que deseamos tome la partida). La ventaja de esto es que la mayoría de los cambios provendrán de la necesidad de añadir cantidades a las partidas que ya existen. Así, si se va a gastar un dinero extra de 15.000 pesetas en reparar el coche (partida que supongamos ya está abierta), todo lo que hay que hacer es recorrer el fichero hasta encontrar ese título e introducir «15000».

Comprobación

Lance el programa y cargue los datos que previamente había grabado en cinta. Ahora llame a la opción 2 desde el menú y compruebe que puede examinar las tres partidas, volviendo al final otra vez al menú. Llame a la opción 2 otra vez, pero esta vez intente añadir o restar alguna cantidad a los dos totales que había introducido previamente. Un nuevo vistazo a las partidas debería revelar que efectivamente sus valores han sido alterados.

Módulo 5.2.9: Supresión de partidas

Una última facilidad a añadir, en relación con las partidas existentes, es la supresión de partidas. En el caso de Contable, el módulo de supresión es más complejo que en los ejemplos anteriores de este tipo. La razón de ello es la existencia de grupos formados en torno a encabezamientos. Mientras que no hay dificultades asociadas a la supresión de partidas únicas o de subtítulos, ¿qué ocurre cuando se suprime un encabezamiento? La respuesta, obviamente, es que no solamente debe desaparecer el encabezamiento en sí, sino todos los subtítulos asociados a él; en caso contrario la cuenta quedaría llena de subtítulos no pertenecientes a ningún encabezamiento, dejándola sin sentido.

Módulo 5.2.9: Líneas 17000 - 17140

```
17000 REM*******************************
17010 REM Supresiones
17020 REM*******************************
17030 pl=i:gr=1
17040 d=0:WHILE LEFT$(a$(hd,pl),1)="*" A
ND d=0:d=1
17050 WHILE LEFT$(a$(hd,pl+gr),1)="$"
17060 gr=gr+1
17070 WEND
17080 WEND
17090 FOR k=pl TO c(hd)-gr-1
17100 a(hd,k)=a(hd,k+gr)
17110 a$(hd,k)=a$(hd,k+gr)
17120 NEXT k
17130 c(hd)=c(hd)-gr
17140 RETURN
```

Comentarios

Línea 17030: La posición en la que va a tener lugar la supresión es enviada desde el módulo anterior en la forma de la variable I. Para los objetivos de este módulo su valor se transfiere a PL. La variable GR (GRupo) registra cuántas partidas hay que suprimir. Inicialmente está puesta a 1, y solamente se incrementará si la partida a suprimir es un encabezamiento con varios subtítulos bajo él.

Líneas 17040-17080: Estos dos bucles anidados son activados solamente si la partida especificada es un encabezamiento. El bucle interno explora las entradas siguientes, contando cuántas de ellas están precedidas por un «$» (indicando por tanto que son subpartidas del encabezamiento). El resultado de dicha cuenta se guarda en GR.

Líneas 17090-17120: Un bucle típico para colapsar una matriz, y suprimir uno de sus elementos. En este caso, la diferencia es que, en vez de copiar el elemento X en el espacio X-1, y por tanto copiar cada elemento un lugar más abajo, los elementos son transferidos GR lugares suprimiéndose así GR elementos, es decir, el número de elementos del grupo formado en torno al encabezamiento.

Comprobación

Siguiendo el procedimiento de comprobación del módulo anterior, debería no solamente poder recorrer las partidas y alterarlas, sino suprimirlas a voluntad. Si suprime la partida rotulada como TITULO, debería observar que SUBTITULO desaparece con ella.

Módulo 5.2.10: Presentación de las cuentas

Después de todas las preparaciones, el módulo que da sentido a la totalidad, presentando la cuenta en su forma final. Al igual que el módulo equivalente de Banquero parece complejo, pero una vez que haya visto la presentación en pantalla, verá rápidamente porqué las cosas se han dispuesto en la forma en que lo están.

Módulo 5.2.10: Líneas 18000 - 18310

```
18000 REM*******************************
18010 REM Presentar cuentas
18020 REM*******************************
18030 tt=0:ss=0
18040 CLS:PRINT "Presentar cuentas:":PRI
NT
18050 PRINT:INPUT "Desea sacar las cuent
as por la impresora (s/n)";p$:ca=0:IF LO
WER$(p$)="s" THEN ca=8
18060 CLS
18070 PRINT #ca,TAB(20-LEN(hd$)/2);hd$:P
RINT
18080 FOR i=0 TO c(hd)-1
18090 tt=tt+a(hd,i)
18100 IF LEFT$(a$(hd,i),1)="*" THEN PRIN
T #ca
18110 IF LEFT$(a$(hd,i),1)="$" THEN PRIN
T #ca,"  ";
18120 PRINT #ca,MID$(a$(hd,i),2);
18130 d=0:WHILE LEFT$(a$(hd,i),1)<>"*" A
ND d=0:d=1
18140 PRINT #ca,TAB(21);
```

```
18150 IF LEFT$(a$(hd,i),1)="%" THEN PRIN
T #ca,TAB(31);
18160 PRINT #ca,USING "#######.##";a(hd,i
);
18170 IF LEFT$(a$(hd,i),1)="$" THEN ss=s
s+a(hd,i)
18180 WEND
18190 PRINT #ca
18200 WHILE ss<>0 AND LEFT$(a$(hd,i+1),1
)<>"$"
18210 PRINT #ca,TAB(21);"----------"
18220 PRINT #ca,TAB(31);USING "#######.##
";ss
18230 ss=0
18240 WEND
18250 NEXT i
18260 PRINT #ca,TAB(31);"----------"
18270 PRINT #ca,"TOTAL:";TAB(31);USING "
#######.##";tt
18280 IF ca=8 THEN RETURN
18290 PRINT:PRINT "Pulse cualquier tecla
 para continuar"
18300 WHILE INKEY$="":WEND
18310 RETURN
```

Comentarios

Línea 18090: La variable TT se utilizará para almacenar el saldo de la cuenta según ésta se va presentando.

Líneas 18100-18120: Estas líneas imprimen el nombre de la partida. Si se trata de un encabezamiento, primero se imprime una línea en blanco que sirve de separación respecto a lo que se ha presentado por encima. Si es un subtítulo, se sangran dos espacios antes de presentarlo.

Líneas 18130-18180: Estas líneas se ocupan de la presentación de las cantidades asociadas a las partidas únicas y a los subtítulos. Las correspondientes a los subtítulos se presentarán a partir de la posición 21 de la línea, y las correspondientes a partidas únicas a partir de la posición 31. Si la partida es un subtítulo, entonces el subtotal de las partidas de ese grupo se guarda provisionalmente en SS.

Líneas 18190-18230: Este bucle opera solamente cuando se está procesando un grupo, lo que está indicado por el hecho de que SS es distinta de 0, y la partida siguiente no forma parte del grupo (es decir el grupo está completo). El efecto del bucle es presentar el total del grupo correspondiente en la columna principal de números (a partir de la posición 31 de la línea).

Comprobación

Vuelva a cargar los datos que había almacenado en cinta y simplemente llame a la opción 3 del menú principal. Si esta prueba es superada con éxito, el programa está listo para ser utilizado.

Otros libros sobre MICROINFORMATICA

C. Prigmore	**BASIC** Curso de autoenseñanza para principiantes
R. Pawson	**El libro del robot**
P. Kuczora/Ch. King	**Introducción al MSX**
G. Ladevie	**La gestión con BASIC** Comercio y pequeña empresa
G. Guérin	**Microinformática de gestión** Alternativas y utilización
A. P. Mullan	**El ordenador en la Educación Básica** Problemática y metodología
D. Daines	**Las bases de datos en la Educación Básica** Utilización y ejemplos
G. W. Orwig/W. S. Hodges	**Programas educacionales para su ordenador personal**
M. D. Segarra/J. Gayán	**LOGO para maestros**
M. J. Winter	**El Cuaderno de LOGO** Ejercicios ilustrados para el Apple
M. J. Winter	**El Cuaderno de LOGO** Ejercicios ilustrados para el Commodore
A. Bork	**El ordenador en la Enseñanza** Análisis y perspectivas de futuro
A. Myx	**LOGO. Tratamiento de listas y palabras**
J. A. Aznar	**Informes de evaluación** Un modelo informático para la Enseñanza
P. Pellier	**Lenguaje máquina del ZX Spectrum** Subrutinas y trucos
T. Hartnell	**Juegos dinámicos para el ZX Spectrum**
R. G. Hurley	**Los Micro Drives del ZX Spectrum** Utilización y aplicaciones
D. Lawrence	**Programas prácticos para el Spectrum** Una biblioteca de módulos y subrutinas
I. Sinclair	**Introducción al Commodore 64**
I. Sinclair	**Lenguaje máquina del Commodore 64**
S. Money	**Gráficos y sonidos para el Commodore 64**
O. Bishop	**Juegos para el Commodore 64**
M. England/D. Lawrence	**Programas en código máquina para el Commodore-64** Gráficos y sonidos
D. Lawrence	**Programas prácticos para el Commodore-64** Una biblioteca de módulos y subrutinas
D. Lawrence/M. England	**Introducción al código máquina del Commodore-16**
J. Billingsley	**Robótica y sensores para el Commodore** Proyectos prácticos para aplicaciones de control
B. Lloyd	**Introducción al Dragon**
D. Lawrence	**Programas prácticos para el Dragon**
K. y S. Brain	**Gráficos y sonidos para el Dragon** Incluye subrutinas en código máquina
K. y S. Brain	**Inteligencia artificial en el Dragon**
I. Sinclair	**Lenguaje máquina del Dragon**
M. James/S. M. Gee/K. Ewbank	**Juegos para el Dragon**
V. Apps	**40 juegos educacionales para el Dragon**
D. Lawrence/S. Lane	**Programas prácticos para el Amstrad** Una biblioteca de módulos y subrutinas
R. Hyde	**Programación en código máquina del Apple II**

Este libro es una colección de programas de aplicación completos. Entre las áreas cubiertas por dichos programas se encuentran el control de ingresos e impuestos, el almacenamiento y recuperación de información, economía doméstica y manejo de agendas, gráficos creativos y técnicas eficaces de presentación visual, música, programas educacionales y una recopilación de programas más pequeños que exploran las posibilidades del Amstrad como reloj, cronómetro y demás aplicaciones horarias.

Todos los programas del libro están claramente explicados y escritos en módulos fácilmente identificables, de forma que puede copiar los métodos expuestos dentro de sus propios programas.

El Amstrad aporta nueva savia a los ordenadores domésticos, en lo referente a capacidad y valor, y ofrece la perspectiva de áreas nuevas y apasionantes de aplicación. Los libros de David Lawrence han proporcionado a miles de personas la posibilidad de escribir sus propios programas de aplicación. En éste nos muestra cómo puede liberarse toda la potencia del Amstrad a través de su nuevo y potente Basic.

David Lawrence es autor de varios libros de gran éxito sobre ordenadores, entre los que se cuentan: *Programas prácticos para el Spectrum, Programas en código máquina para el Commodore 64, Programas prácticos para el Commodore 64, Introducción al código máquina del Commodore 16* y *Programas prácticos para el Dragon,* publicados en la Editorial Gustavo Gili, S. A.

Simon Lane es un programador respetado, cuyos programas de juegos se han vendido en gran cantidad de países de todo el mundo.

Editorial Gustavo Gili, S. A.
Rosellón, 87-89
08029 Barcelona

PROGRAMAS PRACTICOS PARA EL AMSTRAD

D. Lawrence
S. Lane

GG

GG

Franquear
como
Tarjeta
Postal

Editorial Gustavo Gili, S.A.
Apartado de Correos 35.149

08080 Barcelona (España)

SOLICITUD GRATUITA DE CATÁLOGOS

Estimado lector:

Le quedamos muy agradecidos por adquirir este libro, el cual deseamos responda completamente a sus necesidades.
Al devolvernos esta tarjeta solicitando catálogos, le rogamos nos preste su colaboración respondiendo a las siguientes preguntas. Muchas gracias.

¿DE QUÉ OBRA HA RETIRADO ESTA TARJETA? .....

.....

¿CÓMO CONOCIÓ ESTE LIBRO?

☐ Reseña crítica ☐ Anuncio prensa ☐ Escaparate ☐ Folleto
☐ Recomendación personal ☐ Aconsejado por el profesor
☐ .....

¿DÓNDE ADQUIRIÓ ESTA OBRA?

☐ Librería ☐ Vendedor visitador ☐ Directamente de la Editorial
☐ Feria de Libros ☐ .....

Apellidos .....

Nombre .....

Profesión .....

Estudiante de ..... Curso .....

Dirección particular .....

Población ..... D.P. .....

Provincia ..... País .....

Deseo recibir gratuitamente, a vuelta de correo, el catálogo de sus publicaciones, así como información periódica de las novedades que vayan publicando sobre las materias que les señalo:

- ☐ **Arquitectura. Construcción. Urbanismo**
- ☐ **Ingeniería general**
- ☐ **Mecánica**
- ☐ **Plásticos**
- ☐ **Electricidad**
- ☐ **Electrónica**
- ☐ **Informática**
- ☐ **Diseño. Dibujo**
- ☐ **Tecnología y Sociedad**
- ☐ **Fotografía. Cine. Teatro. Televisión**
- ☐ **Comunicación. Mass Media**
- ☐ **Arte. Estética**
- ☐ **Obras de arte numeradas**
- ☐ **Literatura. Diccionarios**
- ☐ **Decoración. Muebles**
- ☐ **Interés general**

AMSTRAD CPC

MÉMOIRE ÉCRITE
MEMORY ENGRAVED
MEMORIA ESCRITA

https://acpc.me/