AÑO II N.º 23

MICROHOBBY

AMSTRAD semanal

REVISTA INDEPENDIENTE PARA USUARIOS DE ORDENADORES AMSTRAD

AÑO II N.º 23

160 Ptas. (Incluido I.V.A.)

Canarias, Ceuta y Melilla 165 ptas.

METODOS DE ORDENACION OPTIMIZADOS

LA OFICINA MODERNA ESTA AQUI: CONTROL DE STOCKS CP/M

Instrucciones de rotación y desplazamiento: ejemplos de aplicación

VENTANA DE COMANDOS INTERACTIVOS CON WINDOWKEY

SOFTWARE

La marca del Zorro en las pantallas de Amstrad

HOBBY PRESS, S.A.

SI BUSCAS LO MEJOR ERBE Software LO TIENE

LA REPLICA AL CELEBRE "ROLLING" DE LAS MAQUINAS

EL JUEGO MAS ADICTIVO QUE PUEDAS ENCONTRAR

GYROSCOPE

DISTRIBUIDOR EXCLUSIVO PARA ESPAÑA ERBE SOFTWARE, SANTA ENGRACIA, 17. Tel.: 447 34 10. DELEGACION BARCELONA, Avd. MISTRAL, 10. Tel. (93) 432 07 31

Director Editorial
José I. Gómez-Centurión
Director Ejecutivo
Víctor Prieto
Subdirector
José María Díaz
Redactora Jefe
Marta García
Diseño
José Flores
Colaboradores
Francisco Portalo, Pedro Sudón
Miguel Sepúlveda,
Francisco Martín,
Jesús Alonso, Pedro S. Pérez
Amalio Gómez
Juan J. Martínez,
David Sapuerta, Alberto Suñer,
Eduardo R. Velasco
Secretaria Redacción
Carmen Santamaría
Fotografía
Carlos Candel
Javier Martínez
Portada
M. Barco
Ilustradores
Javier Igual, J. Pons, F. L.
Frontán, J. Septien, Pejo, J. J.
Mora, Luigi Pérez

Edita
HOBBY PRESS S.A.

Presidente
María Andrino
Consejero Delegado
José I. Gómez-Centurión

Jefe de Publicidad
Concha Gutiérrez
Publicidad Barcelona
José Galán Cortes
Tel: (93) 303 10 22/313 71 62

Secretaria de Dirección
Marisa Cogorro

Suscripciones
M.ª Rosa González
M.ª del Mar Calzada

Redacción, Administración y Publicidad
La Granja, 39
Polígono Industrial de Alcobendas
Tel.: 654 32 11
Telex: 49 480 HOPR

Dto. Circulación
Carlos Peropadre

Distribución
Coedis, S. A. Valencia, 245
Barcelona

Imprime
ROTEDIC, S. A. Crta. de Irún.
Km. 12.450 (MADRID)
Fotocomposición
Novocomp, S.A.
Nicolás Morales, 38-40
Fotomecánica
GROF
Ezequiel Solana, 16
Depósito Legal:
M-28468-1985

MICROHOBBY

AMSTRAD

sumario

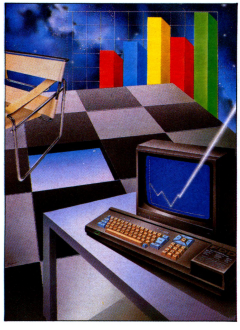
Año II • Número 23 • 4 de Febrero al 10 de Febrero 1986
160 ptas. (incluido I.V.A.)
Precio Canarias, Ceuta y Melilla 155+10 ptas. sobretasa aérea.

5 Primera plana

Aple a por todas. Prolog para ***Amstrad.***

Primeros pasos 6

Después de hablar durante largo tiempo de las matrices, de cómo se crean y de cómo se manejan, es hora de usarlas en una aplicación práctica de uso muy común en informática.

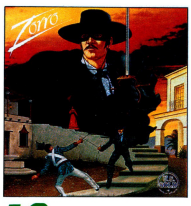


12 Banco de pruebas

La oficina moderna, para gestionar eficazmente algo tan complejo como una empresa, requiere de herramientas cada vez más potentes y sofisticadas. ***Amstrad*** *tiene algo que decir a este respecto.*

Análisis 16

Antes de aplicar un algoritmo de ordenación a algo, no es mala idea prepararlo para que la clasificación tarde el mínimo tiempo posible. Una técnica, no la mejor pero sí la más simple, es el método ***«hash».***



18 Mr. Joystick

El programa se llama ***"El Zorro".*** *No necesita de mucha introducción. Uno de los más populares héroes de todos los tiempos cobra vida en la pantalla del* ***Amstrad.***

Serie Oro 20

Microfruits *es una simulación de una máquina tragaperras que tiene el gran mérito de estar integramente realizada en Basic. Los gráficos merecen ser vistos.*

24 ProgramAcción

El ***Amstrad*** *posee algo muy útil: las teclas de función redefinibles. Pero tienen un inconveniente: hay que acordarse de lo que hemos puesto en cada tecla. Bueno, lo tienen hasta ahora. Programacción os explicará por qué.*

28 Código Máquina

Las instrucciones de rotación y desplazamiento, tal vez una de las más difíciles de dominar y entender, poseen aplicaciones prácticas muy patentes, que justifican su importancia y dificultad.

MICROHOBBY

AMSTRAD semanal

REVISTA INDEPENDIENTE PARA USUARIOS DE ORDENADORES AMSTRAD

LE OFRECE AHORA SUS PROGRAMAS YA GRABADOS PARA QUE VD. NO TENGA QUE TECLEARLOS. TOTALMENTE DESPROTEGIDOS CON EL OBJETO DE FACILITAR SU COPIA EN DISCO.

Todos los programadores y aficionados a la microinformática, sabemos lo tedioso y propenso a errores que resulta el teclear un listado de un programa. Para facilitar tu labor al máximo y evitar que malgastes largas horas sobre el teclado de tu ordenador tratando de descifrar incomprensiblemente mensajes de error.

AMSTRAD SEMANAL te ofrece cada mes los programas publicados en los cuatro números correspondientes, en una cinta de cassette desprotegida, que te permitirá copiar los programas en disco y tener acceso a los listados para su estudio y posterior edición de rutinas.

Programas incluidos en la cinta número 1

Título	Rev. n.	Título	Rev. n.
EASYDRAW	1	MAD ADDER	3
EGOBLITZ	2	HEXER	3
CODIGO SECRETO	2	CHARGEN	4
VENTANAS	2	PROGRAMACION	4
BIORRITMOS	3		

Programas incluidos en la cinta número 2

Título	Rev. n.	Título	Rev. n.
GRAFICOS	5	INCOGNITON	7
MUSICA	6	MONITOR	5
TRON	6	ANALISIS	5-8
ENSAMBLADOR	8	CEDRIC	8
HEXER1	8	ANIMACION1	7
TOOLKIT	8	ANIMACION2	8
PRIMEROS PASOS	7	SMALET	7

Programas incluidos en la cinta número 3

Título	Rev. n.	Título	Rev. n.
ANALISIS	9-12	SPRITE12	11
FRUTAS	9	AMSCARD	11
MENUDISC	9	OTELO	12
RSX101	10	EVENT21	12
RSX102	10	EVENT22	12
CUATRORAYA	10	FRUTTIES	12
SPRITE11	11		

Por sólo 675 pts.
(incluidos gastos de envío)

COMPATIBLES CON LOS MODELOS CPC-464, CPC-664 y CPC-6128

Recíbelos en tu casa cómodamente enviándonos con la menor demora posible, el cupón que se encuentra en la última página de la revista

Primera plana

LENGUAJES IA

Idealogic comercializa en España Le-Lisp, una versión muy potente del Common Lisp para sistemas basados en MS DOS, PC DOS y para el MacIntosh.

Para los IBM y compatibles, la configuración mínima de memoria requerida es de 256 Kbytes. Esta gran cantidad de memoria se debe a que todos los lenguajes de IA la consumen rápidamente, para poder llevar a cabo las complejas operaciones que se les exije.

En el caso del Mac, se necesitan 512 kbytes de RAM; esto se debe probablemente a que el Le-Lisp para el MacIntosh usara la interface de usuario típica, con iconos y ventanas, lo cual consume mucha memoria, junto con que los creadores del lenguaje habrán aprovechado la oportunidad de hacer que se pueda usar toda la memoria disponible, por aquello de la potencia.

El distribuidor clama que su lenguaje es apropiado para prácticamente cualquier aplicación, no solamente las típicas de proceso de palabras propias de la IA.

Se supone que se pueden realizar con toda comodidad cálculos científicos e ingenieriles, además de cosas como sistemas expertos y proceso del lenguaje natural.

Una faceta muy importante de Le-Lisp, y que las implementaciones de este lenguaje comienzan a tener cada vez más a menudo en ordenadores personales, es la existencia de un compilador: una vez creado el programa y depurado mediante un programa de tipo Debugger *(también incluido)*, entra en acción el compilador para tratar de acelerar al máximo la ejecución del programa, pues todas las aplicaciones escritas en intérpretes, y más las de los lenguajes IA, son de ejecución muy lenta.

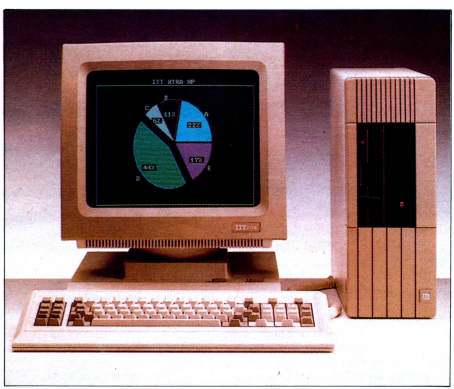
La constante proliferación de lenguajes IA no sólo sucede en el campo de los ordenadores personales; también parece que le ha llegado el turno al **Amstrad**.

La casa Hisoft esta desarrollando para él un lenguaje Prolog, descendiente de Isp y clave del proyecto japonés de la Quinta Generación de ordenadores. El programa se encuentra todavía en estado de prototipo, si bien en un estado avanzado de desarrollo, y todavía no tenemos noticias concretas de sus características y posibilidades. Esperamos que aparezca próximamente en Inglaterra, y desconocemos, también, su fecha de comercialización en nuestro país.

APPLE, A POR TODAS

A parte de la noticia de que el Apple II será compatible con el MacIntosh, suponemos que mediante el uso de alguna interface de usuario similar, tipo WIMP, el bombazo estalló cuando Apple Computer comunicó el próximo lanzamiento de dos nuevos ordenadores tipo Mac, uno para enero de este año y otro para un poco más adelante, dentro también del 86.

Sobre las especificaciones de los nuevos modelos, poco o nada se sabe aún; sin embargo, se especula con que Apple empleará como microprocesador de los nuevos Mac el 68020 de Motorola, una auténtica «salvajada» de microprocesador de 32 bits, junto con la posibilidad de que vengan con mucha más memoria en su configuración base.



ITT XTRA XP

ITT ha creado un nuevo ordenador personal compatible con la gama IBM PC/XT/AT de reducido precio y altas prestaciones.

Está basado, cómo no, en el 80286 de Intel, funcionando a 8 megahertzios, y en su configuración básica posee 512 kbytes de RAM. Como todos los compatibles AT, puede usar el sistema operativo MS DOS y el XENIX, versión del UNIX que permite soportar un entorno de trabajo multiusuario y multitarea.

El sistema puede ampliarse hasta 1,64 Megas de RAM, y existen varias versiones con diferentes configuraciones en lo que se refiere al almacenamiento masivo de datos, es decir, discos duros de 10 y 20 Megas, además del disco floppy de 1,2 Megas.

ITT asegura que su sistema es uno de los más veloces del mercado, teniendo en cuenta el precio.

Dice también que el procesador 80286 es ideal para aplicaciones industriales o de negocios que requieran manejo exhaustivo de gráficos y comunicaciones.

MATRICES EJEMPLARES

Ha llegado la hora de poner la guinda al pequeño «pastel» que hemos ido haciendo mediante elementos sencillos pero muy útiles y que nos habrán dado como resultado un conocimiento bastante extenso sobre una de las herramientas más utilizadas en programación: las matrices.

Javier Igual

Seguramente la mejor forma de llegar a comprender a la perfección toda una serie de conceptos teóricos de cualquier tipo es seguir paso a paso y despacito el desarrollo de un problema práctico en el que se apliquen todas las definiciones, o al menos la mayoría, que les hemos contado anteriormente. Pues **¡vamos a ello!**

Lo primero que debemos hacer es definir y delimitar claramente el problema que queremos resolver. A continuación, y una vez analizado, procedemos al diseño de su resolución dividiendo en pequeños trozos que realizan unas acciones más concretas que la general y, por tanto, será más sencillo para nosotros fabricar un programa que haga lo que queramos en cada una de estas acciones particulares.

Una vez que tengamos solucionados todos estos problemitas, lo único que nos queda es unir convenientemente los modulitos que hemos codificado y formar un **gran programa** que cumpla con todas las especificaciones y ejecute correctamente las acciones que definimos en el **problema general,** es decir: ¡que funcione!

Diseño de un programa

Vayamos, pues, al grano. Nuestro caso va a ser el de un programador que ha de codificar un programa que cree, modifique y haga todo lo que se nos pueda ocurrir con los datos de los empleados de una empresa. **¿Le parece muy complicado?**

¿Qué datos son los que vamos a manejar? Todos los que usted quiero, pero, ¡no se extienda, por favor! Nosotros hemos pensado que con los clásicos nos puede valer. Por un lado tendremos los **particulares:** nombre y apellido, domicilio y además usaremos los **laborales** como pueden ser el número de personal, la referencia o sección de trabajo, la categoría y el sueldo.

Vamos a suponer que la empresa que nos ha pedido este programa ha previsto que como máximo va a tener 100 empleados. ¡Nos imaginamos que todos estamos pensando ya en agrupar los datos de cada tipo en matrices de 100 elementos! ¿No?

Pues sí. Vamos a definir y dimensionar las matrices que van a contener los datos de los empleados de esta empresa. Para ello utilizamos el programa 1.

Programa uno

En él dimensionamos tantos arrays como datos queramos tener de cada empleado. El número máximo del subíndice de uno de sus elementos está muy claro que será 100 ya que éste es el número máximo de empleados previsto.

Los datos de tipo numérico estarán formando parte de matrices de elementos numéricos: los sueldos —números— deben estar contenidos en arrays de este tipo y por eso hemos definido uno tal como:

```
DIM sueldo (100)
```

Las cadenas, o datos literales, se almacenarán en matrices de elementos alfanuméricos: el nombre —cadena de literales— estará incluido en el array dimensionado:

```
DIM nombre$ (100)
```

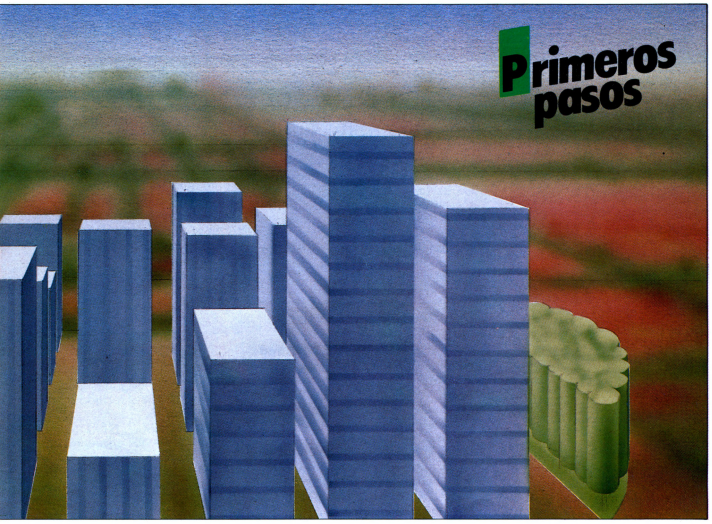
Resumiendo, estructuramos los datos de forma que cada uno de ellos sea un elemento de una matriz numérica o alfanumérica que agrupe todos los de unas determinadas características: sueldos por un lado, calles por otro, etc.

Quizá usted también haya pensado que una vez dimensionados los datos lo más oportuno será definir a continuación qué operaciones queremos realizar con ellos. Pensemos.

La primera será crear un fichero —es obvio **¿no?**—. Una vez creado, el siguiente paso podría ser **incluir** datos. Esta opción se utilizará para dar de alta en la empresa a un nuevo trabajador.

Otra de las posibilidades es que tengamos que dar de baja a un empleado o quizá modificar los datos de otro.

También tendremos que contemplar que se pueda buscar toda la información de una determinada persona.



Por último, los datos contenidos en la memoria no se guardan con el programa al grabar éste en el disco. **¡Horror!** ¿Los perderemos entonces? ¿Tendremos que teclearlos cada vez que ejecutemos el programa?

No se preocupe. Como programadores **avispados** que somos, ustedes y nosotros, ya hemos previsto esta situación. ¡Una de las acciones que realizará este programa es salvar todos los datos en disco o cinta!

Y ¿cómo volver a meterlos en la memoria después de cargar el programa? Ya veremos cómo hacerlo.

Así que parece que nuestro programa tendrá que realizar siete acciones diferentes sobre los datos de los empleados de la empresa.

— CREAR nuevo fichero.
— INCLUIR datos.
— BORRAR.
— MODIFICAR.
— BUSCAR.
— CARGAR datos de cinta/disco.
— SALVAR datos en cinta/disco.

Para dirigir la ejecución del programa para que realice uno de estos cometidos utilizaremos un modulito semejante al programa 2.

Programa dos

En él imprimimos toda la información sobre las opciones que tendrá nuestro programa —líneas 1020 a 1120.

Después pedimos el número de la acción a realizar sobre nuestros datos y lo almacenamos en la variable **«opción»** —línea 1140.

Las líneas 1150 a 1210 son las que dirigen la ejecución del programa a un lugar determinado. Dependiendo que se cumpla o no una condición, el programa hace o no una instrucción.

En nuestro caso, la condición es que si «opción» tiene un valor comprendido entre 1 y 7, saltamos —GOTO...— a una determinada línea. La instrucción GO TO hace que saltemos directamente al número de línea que tiene colocado detrás. Allí continuamos con la ejecución del programa.

Si **opción** no está comprendida entre 1 y 7 —no cumple con las anteriores condiciones— saltamos a la línea 1140 para pedir una nueva opción, esta vez correcta.

Y así es a grandes rasgos como funciona un sistema de menús. Estamos seguros que puede mejorarle y perfeccionarle pero como indicación puede valer éste.

Una vez elegida alguna opción, vamos a intentar ir poco a poco viendo cómo las realizaríamos. ¿En qué consistirá crear un nuevo fichero *(opción 1)*? ¡Elemental, mi querido amigo! Eche una ojeada al programa 3 y lo comprenderá enseguida.

Programa tres

En las líneas 2020 a 2100 borramos de la memoria los contenidos de las matrices en las que hemos almacenado los valores de los datos de los empleados que trabajan en la empresa.

Vemos que la orden ERASE también libera la memoria del espacio que había reservado para ellas. En este momento no existe ninguna de estas matrices para el ordenador.

Han desaparecido junto con lo que contenían.

Pero no tenga cuidado. Las volveremos a definir con las líneas 2110 a 2190 pero esta vez las numéricas *—contienen números—* estarán inicializadas a cero y las alfanuméricas *—contienen literales—* lo estarán con la cadena vacía.

Así pues, ya las tenemos preparadas para volver a contener nuevos datos y hemos visto cómo se pueden redefinir.

Con la línea 2200 volvemos al menú principal que nos pedirá otra nueva opción.

Otra de las acciones a realizar es INCLUIR datos en el fichero *(opción 3)*. **¿De qué se trata?** Muy fácil. Cuando se da de alta un trabajador en la empresa tenemos que almacenar todos sus datos en cada una de las matrices que hemos definido para contenerlos.

Mediante el programa 4 realizaremos esta función:

Programa cuatro

Con las líneas 3040 a 3120 introducimos los nuevos valores dentro de cada una de las matrices en el espacio reservado para el elemento indicado por el contenido de la variable **índice.**

Pero, **¿cómo determinamos el lugar exacto en el que debemos hacerlo?** ¿Se atreve a contestarnos? Al crear el fichero, el elemento cero de la matriz que contiene los números de personal de los empleados, **numpers,** tiene el valor 0 —se inicializa a 0 como el resto de los que componen la tabla.

Si cada vez que incluimos el conjunto de datos de un trabajador aumentamos este elemento en uno, podemos utilizarle para almacenar en el número de empleados cuyos datos ya están metidos dentro del fichero.

Es evidente que si contiene el último elemento lleno de la matriz, al sumarle 1 tenemos el valor del índice del siguiente elemento de la tabla, es decir, el primero que está vacío.

Por tanto, en la línea 3030 hacemos que la variable **índice** contenga el número de elementos de la tabla donde tenemos que incluir los nuevos datos.

No olvidemos que nos queda actualizar el número de elementos llenos. Para ello, tenemos la línea 3130 que es un alarde de obediencia, se encarga de ello y nos deja preparado este número —**numpers** (0)— para inclusiones posteriores.

De modo análogo a los casos anteriores, la línea 3140 nos envía otra vez al menú principal.

Como ve de nuevo utilizamos el elemento cero de una matriz para contener el número de elementos de la misma que están llenos. Recuerde que es de bastante utilidad.

Ya que sabemos incluir datos dentro de un fichero, vamos ahora a realizar la operación contraria: BORRAR datos. Esencialmente consiste en dar de baja todos los datos correspondientes a un determinado empleado. Hagámoslo con el programa 5.

Programa cinco

En él primeramente el **Amstrad** nos pide el nombre del empleado cuyos datos queremos borrar —línea 4030.

Por medio del bucle WHILE... WEND —líneas 4050 a 4070— buscamos el índice que tiene el elemento que contiene el nombre a borrar. Inicializamos la variable «contador» a 1 —línea 4040— e incrementamos su valor en 1 cada vez que se ejecute el bucle anterior.

Cuando se cumpla la condición de salida —que el nombre que hemos metido en **nombre$** sea igual que el de **nombre$** *(contador)*— la variable **contador** tendrá el número del índice del elemento que queremos borrar.

Si a partir de este elemento sustituimos el valor que contiene por el del siguiente, conseguimos borrar el que queríamos y a la vez correr todos los valores que le siguen una posición hacia delante.

Si esto lo hacemos para todas y cada una de las matrices borraremos todos los elementos que tienen por subíndice la variable **contador** y correremos una posición todos los siguientes para «rellenar» el hueco dejado.

La ejecución del bucle FOR... NEXT de las líneas 4080 a 4180 nos evita tener que hacerlo manualmente. Como límite inferior tiene **contador** —empezamos así en el elemento borrador— y como límite superior el número de elementos llenos **«numpers** *(0)*».

Solamente nos queda que la línea 4190 nos actualice el número de elementos llenos restándole 1 a **«numpers** *(0)*». Con la línea 4200 volvemos al menú principal.

Para MODIFICAR un dato seguimos el siguiente camino:

— Preguntar el nombre a modificar.

— Buscar en las tablas los datos del nombre que hemos dado.

— Si son los que queremos modificar, modificarlos.

— Si no lo son, buscar el siguiente dato que tenga el mismo nombre y una vez encontrado repetir el ciclo.

— Suponiendo que no lo encontráramos, sacaríamos el mensaje de **NO EXISTE.**

— Después de todo esto volveríamos al menú principal.

Veamos cómo lo hace el programa 6.

Programa seis

La línea 5030 asigna el valor que introducimos por el teclado a la variable **nombre$.**

El bucle FOR... NEXT —líneas 5050 a 5080— nos encuentra el índice de los elementos cuyo nombre es igual al que queremos modificar. Su valor es el contenido en la variable «índice» menos 1.

La línea 5100 nos presenta en la pantalla los datos correspondientes al nombre buscado.

Si son estos los datos que queremos modificar, procedemos a cambiarlos mediante las instrucciones INPUT de las líneas 5110 a 5220. Observe que utilizamos el índice encontrado anteriormente *—índice menos 1—* para insertar los nuevos valores en los elementos de las matrices que han de contener todos los datos correspondientes al nombre que metimos en la línea 5030.

Cuando el nombre que nos escriba la línea 5100 no es el que queremos, volvemos a buscar otro nombre igual pero esta vez la variable de control del bucle comenzará con un valor en «índice» que es el del elemento que sigue al último que ha salido en la pantalla.

Una vez encontrado un nombre válido repetiremos el proceso anterior para modificar los datos.

Si no encontramos el nombre deseado, la línea 5090 sacará el mensaje **NO EXISTE** con un bucle de retardo para poder leerlo.

Una vez más, al terminar la operación volvemos al bucle principal *—línea 5230.*

Analice todo este proceso y verá que es mucho más sencillo de lo que parece.

Desde la línea 6000 a la 6120, le aseguramos que el camino recorrido es el mismo que en la operación anterior.

La única diferencia entre estos dos módulos estriba en que ahora una vez encontrado el nombre válido sacamos en la pantalla los elementos de cada una de las matrices con un subíndice que será el valor de la variable **«índice»** menos 1.

Estos elementos formarán el conjunto de todos los datos que van asociados al nombre que buscamos.

Para ver esta diferencia eche una ojeada al programa 7 y comprenderá lo que decimos.

Programa siete

Solamente nos queda ahora ver la forma de cargar los datos desde la cinta *(o disco)* y de cómo salvarlos —o guardarlos— para su posterior utilización.

Teniendo en cuenta que estamos dando los **«Primeros pasos»** informáticamente hablando, en estas dos últimas operaciones nos van a aparecer varias instrucciones nuevas. No se preocupe. Ahora les contaremos de una forma muy general su funcionamiento y más adelante las veremos con profundidad cuando lleguemos a los artículos basados en tratamiento de ficheros.

De momento crea lo que le decimos sobre estas instrucciones aunque ya sabe que le dejamos que libremente haga pruebas y practique por su cuenta todo lo relacionado con estos temas.

Para CARGAR en la memoria una serie de datos desde cinta o disco utilizaremos siempre programas más o menos complicados, pero semejantes al siguiente programa 8.

Programa ocho

Su modo de funcionamiento es muy sencillo. En la línea 7030 el ordenador nos pide el nombre del fichero que queremos cargar.

En la línea 7040 nos aparece una de las nuevas instrucciones de las que le hemos hablado:

OPEN IN nomfich$

Su cometido es dejar preparado el fichero de nombre en el contenido en la variable **«nomfich$»** para que nuestro programa pueda leer datos de él.

Esta lectura la hacemos mediante el bucle FOR... NEXT comprendido entre las líneas 7050 y 7150. Las instrucciones INPUT toman los datos desde la cinta (o disco) —#9, que significa canal 9— y las van almacenando en los diferentes elementos de cada una de las matrices que hemos dimensionado.

Una vez leídos todos los datos cerramos el fichero:

CLOSE IN

en la línea 7160 y sacamos un mensaje de fin de lectura —línea 7170.

Con el programa 9 vemos que la forma de salvar los datos en la cinta o en el disco se realiza de una forma similar.

Con la línea 8040 preparamos el fichero pero esta vez para escribir en él los datos que hay almacenado en la memoria.

Las líneas comprendidas entre el bucle FOR... NEXT —líneas 8050 a 8150— son las que se encargan de salvar cada uno de los valores de los elementos de las matrices *(WRITE, #9, variable que significa que el* **Amstrad** *va a grabar en el fichero de cinta o disco el valor que esté almacenado en* **variable**).

En la línea 8160 cerramos el fichero y sacamos el mensaje de **«DATOS SALVADOS»** con la línea 8170.

Bueno, ya parece que está todo. Solamente le queda mezclar todos estos módulos *(programas)* y obtendrá un magnífico archivo de datos de empleados que cualquier gran empresa le quitará de las manos. *(Es broma).*

Nuestra intención es que con este ejemplo le hayan quedado más claras todas sus posibles dudas sobre el uso y abuso de las matrices.

Nos conformamos con que entienda sin problemas que las matrices nos sirven para organizar datos de un mismo tipo bajo un solo nombre; que las tablas pueden tener varias dimensiones *(o subíndices)* siendo su manejo semejante a las de una dimensión; que los elementos de una matriz *(variables con subíndice)* pueden usarse como si fueran variables normales y que para acceder a uno de estos elementos lo hacemos a través de uno o varios subíndices.

Esperamos que esto haya sido suficiente para abrirnos las puertas a la utilización e investigación de esta maravillosa herramienta. Hasta pronto.

Desde que **AMSTRAD Semanal** vio la luz, en septiembre del 85, ha pasado ya una respetable cantidad de tiempo: llevamos juntos 5 meses.

A lo largo de toda nuestra singladura, hemos procurado siempre ofrecer a nuestros lectores una información caracterizada por dos palabras: actualidad e interés, tratando de hacer a **AMSTRAD Semanal** accesible y amena para todos los usuarios o posibles usuarios de ordenadores **Amstrad** con cualquier nivel de destreza. En este sentido, creemos que siempre hay que tratar con particular atención a los que empiezan, porque normalmente son dueños de una gran ilusión y se enfrentan con lo que parecen dificultades insuperables.

Esa fue, y es, la razón de ser de una de nuestras secciones fijas, la de **Primeros pasos,** en la que se pretende guiar al usuario no iniciado desde el principio mismo del arte de programar, hasta las inaccesibles cimas *(eso dicen)* que conllevan el convertirse en amo absoluto del ordenador.

Ahora bien, no todas las personas han seguido, por las razones que sean, nuestra revista desde el primer número, y ahora, o hace poco tiempo, o dento de poco tiempo, si que se decidirán a llevársela a casa, al menos para ver cómo es, y lo que cuenta.

Pensamos que gran parte de estas personas pueden estar llenas de interés por aprender el manejo y las entretelas de su ordenador, y claro, se encuentran con una sección de **Primeros pasos** que lleva funcionando veintitantos números y presupone en el lector, a pesar de su sencillez y claridad, unos ciertos conocimientos, impartidos en artículos anteriores.

Por este motivo hemos decidido acudir en ayuda de ellos, publicando a partir del número siguiente un resumen de lo más importante y crucial de los artículos de **Primeros pasos** de los anteriores números. Así, los interesados podrán ponerse al día rápidamente.

La extensión de esta recapitulación, lógicamente será menor que la de los artículos **normales;** no queremos pecar de redundantes ni ocupar el espacio de páginas necesario para hablar de otros muchos temas, muy interesantes para todos los lectores, sin excepción.

Por tanto, ponemos en marcha este proyecto, acerca del cual nos sería muy útil recibir vuestra opinión, sugerencias, ideas y, sobre todo, constructivas críticas. Como siempre, sabemos positivamente que con vuestra colaboración, podremos hacerlo mucho mejor.

Primeros pasos

```
10 REM PROGRAMA I
20 REM DIMENSIONADO DE MATRICES
30 DIM nombre$(100)
40 DIM apellido$(100)
50 DIM calle$(100)
60 DIM numero(100)
70 DIM localidad$(100)
80 DIM numpers(100)
90 DIM referencia(100)
100 DIM categoria$(100)
110 DIM sueldo(100)
```

```
1000 REM PROGRAMA II
1010 REM MENU PRINCIPAL
1020 CLS
1030 PRINT TAB(10);"***MENU PRINCIP
AL***"
1040 PRINT
1050 PRINT TAB(8);"1.- CREAR NUEVO
FICHERO"
1060 PRINT TAB(8);"2.- INCLUIR DATO
S"
1070 PRINT TAB(8);"3.- BORRAR DATOS
"
1080 PRINT TAB(8);"4.- MODIFICAR DA
TOS"
1090 PRINT TAB(8);"5.- BUSCAR DATOS
"
1100 PRINT TAB(8);"6.- CARGAR DATOS
"
1110 PRINT TAB(8);"7.- SALVAR DATOS
"
1120 PRINT
1130 PRINT TAB(8);
1140 INPUT "INTRODUCE UNA OPCION: "
,opcion
1150 IF opcion=1 THEN GOTO 2000
1160 IF opcion=2 THEN GOTO 3000
1170 IF opcion=3 THEN GOTO 4000
1180 IF opcion=4 THEN GOTO 5000
1190 IF opcion=5 THEN GOTO 6000
1200 IF opcion=6 THEN GOTO 7000
1210 IF opcion=7 THEN GOTO 8000
1220 GOTO 1140
```

```
2000 REM PROGRAMA III
2010 REM CREAR FICHERO
2020 ERASE nombre$
2030 ERASE apellido$
2040 ERASE calle$
2050 ERASE numero
2060 ERASE localidad$
2070 ERASE referencia
2080 ERASE numpers
2090 ERASE categoria$
2100 ERASE sueldo
2110 DIM nombre$(100)
2120 DIM apellido$(100)
2130 DIM calle$(100)
2140 DIM numero(100)
2150 DIM localidad$(100)
2160 DIM numpers(100)
2170 DIM referencia(100)
2180 DIM categoria$(100)
2190 DIM sueldo(100)
2200 GOTO 1000
```

```
3000 REM PROGRAMA IV
3010 REM INCLUIR DATOS
3020 CLS
3030 indice=numpers(0)+1
3040 INPUT"NOMBRE: ",nombre$(indic
e)
3050 INPUT "APELLIDO: ",apellido$(i
ndice)
3060 INPUT "CALLE: ",calle$(indice)
3070 INPUT "NUMERO: ",numero(indice
)
3080 INPUT "LOCALIDAD: ",localidad$
(indice)
3090 INPUT "NUMERO DE PERSONAL: ",n
umpers(indice)
3100 INPUT "REFERENCIA: ",referenci
a(indice)
3110 INPUT "CATEGORIA: ",categoria$
(indice)
3120 INPUT "SUELDO: ",sueldo(indice
)
3130 numpers(0)=indice
3140 GOTO 1000
```

```
4000 REM PROGRAMA V
4010 REM BORRAR DATOS
4020 CLS
4030 INPUT "NOMBRE A BORRAR: ",nomb
re$
4040 contador=1
4050 WHILE nombre$<>nombre$(contado
r)
4060 contador=contador+1
4070 WEND
4080 FOR i=contador TO numpers(0)
4090 nombre$(i)=nombre$(i+1)
4100 apellido$(i)=apellido$(i+1)
4110 calle$(i)=calle$(i+1)
4120 numero(i)=numero(i+1)
4130 localidad$(i)=localidad$(i+1)
4140 numpers(i)=numpers(i+1)
4150 referencia(i)=referencia(i+1)
4160 categoria$(i)=categoria$(i+1)
4170 sueldo(i)=sueldo(i+1)
4180 NEXT i
4190 numpers(0)=numpers(0)-1
4200 GOTO 1000
```

```
5000 REM PROGRAMA VI
5010 REM MODIFICAR DATOS
5020 CLS
5030 INPUT "NOMBRE A MODIFICAR: ",n
ombre$
5040 indice=1
5050 FOR i=indice TO 100
5060 indice=indice+1
5070 IF nombre$=nombre$(indice-1) T
HEN GOTO 5100
5080 NEXT i
5090 IF indice>100 THEN PRINT"NO EX
ISTE":FOR retardo=1 TO 1000:NEXT re
tardo:GOTO 1000
5100 PRINT nombre$(indice-1);" ";ap
ellido$(indice-1)
5110 INPUT "ES ESTE NOMBRE";correct
o$
5120 IF correcto$="no" THEN GOTO 50
50
5130 PRINT"NUEVOS DATOS"
5140 INPUT "NOMBRE: ",nombre$(indic
e-1)
5150 INPUT "APELLIDO: ",apellido$(i
ndice-1)
5160 INPUT "CALLE: ",calle$(indice-
1)
5170 INPUT "NUMERO: ",numero(indice
-1)
5180 INPUT "LOCALIDAD: ",localidad$
(indice-1)
5190 INPUT "NUMERO DE PERSONAL: ",n
umpers(indice-1)
5200 INPUT "REFERENCIA: ",referenci
a(indice-1)
5210 INPUT "CATEGORIA: ",categoria$
(indice-1)
5220 INPUT "SUELDO: ",sueldo(indice
-1)
5230 GOTO 1000
```

```
6000 REM PROGRAMA VII
6010 REM BUSCAR DATOS
6020 CLS
6030 INPUT "NOMBRE A BUSCAR: ",nomb
re$
6040 indice=1
6050 FOR i=indice TO 100
6060 indice=indice+1
6070 IF nombre$=nombre$(indice-1) T
HEN GOTO 6100
6080 NEXT i
6090 IF indice>100 THEN PRINT"NO EX
ISTE":FOR retardo=1 TO 1000:NEXT re
tardo:GOTO 1000
6100 PRINT nombre$(indice-1);" ";ap
ellido$(indice-1)
6110 INPUT "ES ESTE NOMBRE";correct
o$
6120 IF correcto$="no" THEN GOTO 60
50
6130 PRINT"NOMBRE: "
6140 PRINT nombre$(indice-1);" ";ap
ellido$(indice-1)
6150 PRINT"DOMICILIO:"
6160 PRINT calle$(indice-1);" nun."
;numero(indice-1)
6170 PRINT"LOCALIDAD:"
6180 PRINT localidad$(indice-1)
6190 PRINT"REF.: ";referencia(indic
e-1)
6200 PRINT"NUMERO DE PERSONAL: ";nu
mpers(indice-1)
6210 PRINT"CATEGORIA: ";categoria$(
indice-1)
6220 PRINT"SALARIO: ";sueldo(indice
-1)
6230 FOR retardo=0 TO 10000:NEXT re
tardo
6240 GOTO 1000
```

```
7000 REM PROGRAMA VIII
7010 REM CARGAR DATOS
7020 CLS
7030 INPUT "NOMBRE DEL FICHERO: ",n
omfich$
7040 OPENIN nomfich$
7050 FOR i=0 TO 100
7060 INPUT #9,nombre$(i)
7070 INPUT #9,apellido$(i)
7080 INPUT #9,calle$(i)
7090 INPUT #9,numero(i)
7100 INPUT #9,localidad$(i)
7110 INPUT #9,numpers(i)
7120 INPUT #9,referencia(i)
7130 INPUT #9,categoria$(i)
7140 INPUT #9,sueldo(i)
7150 NEXT i
7160 CLOSEIN
7170 PRINT TAB(8);"***DATOS CARGADO
S***"
7180 FOR retardo=1 TO 1000:NEXT ret
ardo
7190 GOTO 1000
```

```
8000 REM PROGRAMA IX
8010 REM SALVAR DATOS
8020 CLS
8030 INPUT "NOMBRE DEL FICHERO: ",n
omfich$
8040 OPENOUT nomfich$
8050 FOR i=0 TO 100
8060 WRITE #9,nombre$(i)
8070 WRITE #9,apellido$(i)
8080 WRITE #9,calle$(i)
8090 WRITE #9,numero(i)
8100 WRITE #9,localidad$(i)
8110 WRITE #9,numpers(i)
8120 WRITE #9,referencia(i)
8130 WRITE #9,categoria$(i)
8140 WRITE #9,sueldo(i)
8150 NEXT i
8160 CLOSEOUT
8170 PRINT TAB(8);"***DATOS SALVADO
S***"
8180 FOR retardo=1 TO 1000:NEXT ret
ardo
8190 GOTO 1000
```

...descubre el N.º 3

ya está en tu quiosco

AMSTRAD

también disponible para

SPECTRUM, PLUS, 128

COMMODORE 64

CARNIVAL

Si te gusta el tiro al blanco, con este programa podrás practicar sin necesidad de salir de casa. Si tienes buena puntería obtendrás disparos gratis.

BLOCKER

Demuestra tu habilidad esquivando las paredes y a tus enemigos. Cuantos más destruyas, más aparecerán ante ti.

SPACE

Al cargar este programa aparecerá ante ti un batallón de alienígenas. Tu misión es destruirlos, pero cuidado, su intención es eliminarte lo antes posible.

HAUNTED

En este caso debes coger todos los puntos que aparecen en el laberinto. ¡Atención!, los fantasmas están enfadados e intentarán deshacerse de ti a toda costa.

VAMPIRO

Es un programa en el cual pueden participar dos jugadores. La misión de cada uno será pintar las lápidas de un color distinto. La destrucción del enemigo significa la victoria.

SPLIT

Es una rutina en código máquina, que te permitirá siete colores en pantalla en MODO 1, en el cual normalmente sólo se pueden utilizar cuatro.

795 pts. (Incluido IVA)

Si no lo encontrara en su quiosco, solicítelo directamente a nuestra editorial.

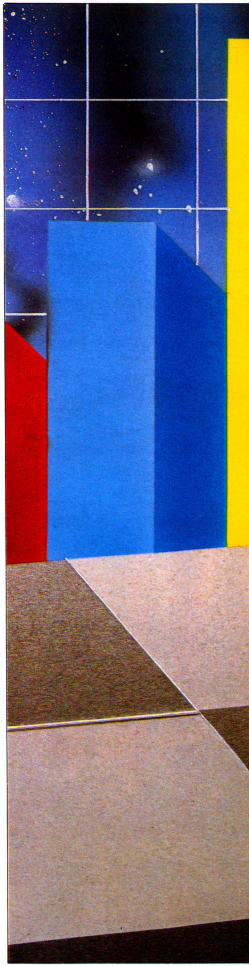
SINTAX, S.A.

Paseo de la Castellana, 268.
28046 Madrid. Tel. (91) 733 25 99

YOUR COMPUTER

La mejor selección de programas de juegos y utilidades, publicados en la revista de mayor difusión de ordenadores de Europa. Ahora reproducidos en cassette, en auténtica exclusiva mundial.

CONTROL DE STOCKS CP/M



M. Barco

Con idea de ofrecer al usuario de ordenadores Amstrad una panorámica lo más amplia y clara posible de los programas de gestión disponibles para la serie CPC, y continuando con la misma política emprendida desde hace algunos números, Banco de Pruebas analiza en esta ocasión un programa que sin duda es un clásico dentro de la gestión y la «papeleta» que resuelve, puede ahorrar horas de esfuerzo y quebraderos de cabeza a mucha gente. Nos referimos a un programa de CONTROL DE STOCKS, funcionando bajo el sistema operativo CP/M, lo cual permite aprovechar al máximo la capacidad de las unidades de disco en el tratamiento de la información.

Como todos los interesados que estén leyendo este artículo saben, un programa de control de stocks resuelve el problema de la compleja gestión de las existencias de un almacén completamente. Por dar una definición de este tipo de programas podríamos decir que: permiten llevar el control de un almacén, conocer en cualquier momento el inventario y obtener un listado de entradas y salidas, por pantalla y/o impresora, que han tenido lugar en determinado período de tiempo.

El estudio de Control de Stocks se abordará de la misma forma que la de los precedentes en esta misma sección: trataremos de dar una detallada descripción del mismo pensando en que a los posibles compradores les interesará saber las posibilidades del mismo.

Queda por comentar un agradable detalle: puede usarse con una o dos unidades de disco indistintamente.

Este programa le permite llevar el control de un almacén, conocer en cualquier momento el inventario y obtener un listado de entradas y salidas que se han producido en un período de tiempo dado.

Al cargarse el programa de STOCK, le aparecerá en pantalla el siguiente menú:

PROCESOS DEL FICHERO
ENTRADAS EN EL ALMACEN
SALIDAS DEL ALMACEN
STOCK EN EL ALMACEN
STOCK BAJO MINIMOS
SALIDAS POR IMPRESORA
FIN DE LA SESION DE TRABAJO

Para escoger cualquiera de estas opciones, con las flechas del cursor elija una de ellas y pulse la tecla **«ENTER»**.

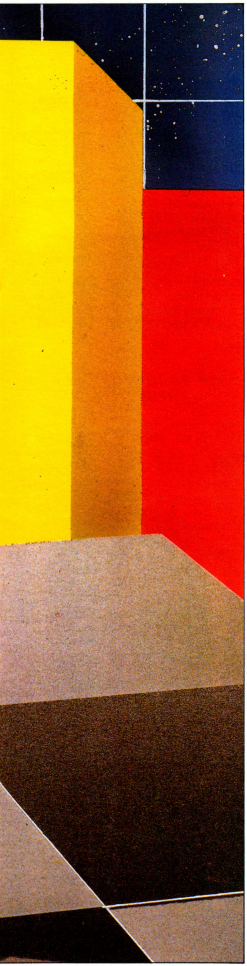
A continuación describiremos con detalle la función y la manera de operar con cada una de las opciones que han aparecido en el menú principal.

PROCESOS DE FICHEROS. En esta opción del menú principal, se encuentran reunidas todas las operaciones que tienen que ver con la creación del fichero de existencias así como su actualización, renovación, corrección, etc.

Cuando Vd. elije esta opción aparece en pantalla el siguiente menú:

INICIALIZAR TODOS LOS FICHEROS
INTRODUCIR NUEVOS ARTICULOS
CORREGIR DATOS DEL FICHERO
CORREGIR FICHERO DE ENTRADAS
CORREGIR FICHERO DE SALIDAS
FIN DE LOS PROCESOS

INICIALIZAR TODOS LOS FICHEROS. Esta opción es la que se utiliza



cuando los discos de datos se usan por primera vez. El proceso inicializa los ficheros de datos preparando los discos para poder recibir los apuntes que Vd. realice.

Antes de ejecutar el proceso, el programa le avisa de que si hay algún fichero antiguo en el disco, éste se destruirá, perdiendo su información. Si Vd. desea salir de esta opción sin ejecutarla pulse la tecla **«ESC»**. Esta tecla *(ESCAPE)* sirve para salir de cualquier opción y volver al menú principal.

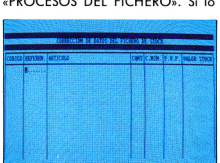
Una vez que el programa ha inicializado los ficheros, todavía le permite la opción de recuperar los que había en el disco mediante otro mensaje de aviso. Si pulsa la tecla **«ESC»**, volverá a tener los ficheros de origen, con cualquier otra tecla volverá al menú de **«PROCESOS DEL FICHERO»**.

INTRODUCIR NUEVOS ARTICULOS. Con esta opción Vd. podrá introducir en el disco el inventario inicial y los nuevos artículos que lleguen al almacén.

El formato de pantalla, representa una hoja donde Vd. apuntará la referencia de su artículo *(8 caracteres)*, la descripción *(30 caracteres)*, la cantidad que hay en stock *(4 caracteres)*, la cantidad mínima de stock *(4 caracteres)* y el precio de venta al público o el precio de coste *(según quiera el inventario por valor de existencias o por valor de venta)*, con 6 caracteres.

Al ir introduciendo los diferentes artículos, el programa asigna a cada uno de ellos un código interno, que servirá para su rápida búsqueda por el programa en otros procesos. Si trabaja con el código que da el programa, los datos se mostrarán en pantalla casi instantáneamente, mientras que si lo hace por referencias de los artículos tardará un poco más.

Si pulsa la tecla **«ESC»** cuando el programa le pide la referencia de un artículo, se interrumpirá el proceso y volverá el control al menú de «PROCESOS DEL FICHERO». Si lo



Banco de pruebas

hace en cualquiera de las demás entradas, se borrará y le preguntará la anterior; es decir, si cuando aparece el cursor en la columna de P.V.P. se pulsa la tecla **«ESC»**, aparecerá el cursor en la columna anterior C. MIN. (cantidad mínima).

CORREGIR DATOS DEL FICHERO. Con esta opción, puede corregir datos del fichero de existencias, precio de venta. El ordenador le preguntará si desea corregir por códigos o por referencias. Si escoge la opción de códigos el acceso al dato será inmediato, en otro caso llevará algo de tiempo. Una vez escogido el método de corrección, el programa le preguntará el código o la referencia del artículo en cuestión, presentando en pantalla la ficha correspondiente a dicho artículo. Pulsando la tecla **«ENTER»**, podrá saltar directamente al dato a corregir y una vez en dicho dato, puede utilizar las teclas del cursor para modificar cualquier caracter. Una vez modificado dicho caracter, pulsando la tecla **«ENTER»**, se introducirá en memoria y debe pulsar de nuevo la tecla **«ENTER»** hasta llegar al final del dato para que se introduzca la modificación en el disco. Una vez modificado un dato entero, el programa le vuelve a pedir un nuevo código o una nueva referencia. Si en ese momento Vd. pulsa la tecla **«ESC»** volverá al menú de **«PROCESOS DEL FICHERO»**.

CORREGIR FICHERO DE ENTRADAS. Con esta opción podrá modificar cualquier dato del fichero de entradas, siendo su funcionamiento idéntico al anterior.

CORREGIR FICHERO DE SALIDAS. El mismo proceso, pero esta vez, con el fichero de salidas de almacén.

FIN DE LOS PROCESOS. Esta opción devolverá el control al menú principal. También puede volver a dicho menú, pulsando la tecla «ESC».

Entradas en el almacén

En este proceso del menú principal, se encuentran englobadas todas

las operaciones que tengan que ver con la entrada de existencias en el almacén, balances de entradas entre dos fechas, entradas de un artículo, entradas de una familia de artículos, etc. Si Vd. elije esta opción, el programa responde con otro menú:

INTRODUCIR DATOS
ENTRADAS ENTRE DOS FECHAS
INICIALIZAR FICHERO DE ENTRADAS
TERMINAR PROCESO

INTRODUCIR DATOS. Esta opción le permite introducir las entradas de artículos ya referenciados en el almacén, actualizando el stock de dicho artículo. Si es la primera vez que la utiliza en el día, el programa le preguntará fecha; una vez introducida, permanecerá en memoria hasta que se apague el ordenador, utilizándola para todos los demás procesos de entradas o salidas que tengan lugar en dicho día. El proceso de introducción de datos es similar a los anteriormente explicados.

ENTRADAS ENTRE DOS FECHAS. Esta opción le permitirá obtener un listado de las entradas que han tenido lugar entre dos fechas cualesquiera. El programa preguntará entre qué dos fechas queremos que saque el listado y de qué referencia. Si cuando el programa pregunta la referencia, se pulsa la tecla «ENTER», obtendremos un listado de todas las entradas que han tenido lugar entre dichas fechas. Si introducimos una referencia, obtendremos únicamente las entradas que han tenido lugar de dicha referencia. Si introducimos parte de dicha referencia, obtendremos un listado de todos los artículos que tengan en común dicha parte de la referencia.

INICIALIZAR FICHERO DE ENTRADAS. Este proceso se utiliza para dejar espacio en el disco y poder seguir introduciendo datos en él. Se utiliza únicamente cuando tenemos muchas entradas o salidas de stock. En el disco, hay espacio aproximadamente para unos 5.000 apuntes de entrada o salidas. Antes de utilizar esta opción, es conveniente sacar un listado en impresora de todas las entradas y salidas del disco o bien hacer una copia del mismo para poder efectuar los balances necesarios en cualquier momento.

El fichero donde se archivan las entradas es el mismo en el que se archivan las salidas del almacén, por lo tanto, si inicializamos el fichero de entradas, también inicializaremos el fichero de salidas. Si se quiere salir de esta opción antes de borrar el tichero, se puede pulsar la tecla «ESC». Pulsando cualquier otra tecla, borraremos el fichero.

TERMINAR PROCESO. Con esta opción retornaremos al menú principal. También se puede volver a dicho menú pulsando la tecla «ESC» en cualquier momento.

Salidas del almacén

Este proceso es idéntico al de entradas en el almacén, pudiéndose trasladar aquí la misma metodología anterior.

Stock en el almacén

Con esta opción, podemos conocer tanto el total del inventario en cualquier momento, así como el inventario de una referencia o el inventario de una familia de referencias.

El programa presenta en pantalla una hoja igual que la que presenta en la introducción de nuevos artículos. En primer lugar nos pregunta por el código del artículo del cual queremos conocer su stock. Si en dicho momento se pulsa la tecla «ESC», retornaremos al menú principal. Si se pulsa la tecla «ESC», retornaremos al menú principal. Si en esta nueva pregunta se pulsa de nuevo la tecla «ENTER» obtendremos un listado de todos los artículos que hay en el almacén y al final el total del inventario.

Stock bajo mínimo

Esta opción le permite conocer todas las referencias que se encuentran por debajo del stock mínimo que Vd. fijó al introducir dicha referencia. Asimismo permite conocer qué referencias de una familia se encuentren bajo mínimos.

Cuando elije esta opción, el programa le pregunta si quiere el listado por pantalla o por impresora. Una vez escogido el periférico adecuado, el programa le preguntará la referencia. Pulsando en este momento la tecla **«ENTER»**, aparecerán en pantalla todas las referencias o artículos por debajo de su stock mínimo. Si en lugar de pulsar la tecla **«ENTER»**, Vd. introduce una referencia o una parte de la referencia, obtendrá un listado de todas las referencias que tengan dicha parte en común y estén bajo mínimos. Durante el proceso de listado, si Vd. pulsa la tecla **«ESC»**, se terminará el proceso pudiendo volver de este modo al menú principal.

Salidas por impresora

Esta opción del menú principal, le permite obtener los listados por impresora de cualquiera de las opciones que aparecen en el sub-menú:

STOCK EN EL ALMACEN
ENTRADAS ENTRE DOS FECHAS
SALIDAS ENTRE DOS FECHAS
IMPRESION DEL STOCK BAJO MINIMOS
FIN DEL PROCESO

Cada una de estas opciones ya se han explicado previamente con lo que puede Vd. utilizar el mismo método que para las salidas por pantalla.

Fin de la sesión de trabajo

Esta opción devuelve el control al CP/M. Si Vd. utiliza esta opción con los discos de datos introducidos en los drives, si éstos no poseen el sistema operativo grabado, puede borrarse la información contenida en dicho disco; por lo tanto, antes de salir del menú principal, se aconseja sacar los discos de sus correspondientes drives e introducir el disco del sistema o un disco que ya posea el sistema operativo. También aconsejamos sacar una copia de los discos de datos regularmente para mayor seguridad, ya que un fallo de tensión mientras se está trabajando, puede provocar el borrado accidental de los discos. Para hacer una copia de los discos de datos, utilice el programa COPYDISC o DISCOPY que viene en el disco del sistema.

FICHA DEL PROGRAMA CONTABILIDAD

Comercializado por: **Indescomp.**
Sistema operativo: **CP/M** 2.2.
Soporte: **Disco**
Funciona con: una o dos unidades de disco.
Precio: **12.000** ptas.
Compatible con:
CPC 464, 664, 6128.

Análisis

ORDENACION HASH

Vamos a estudiar en el presente análisis un método muy usado en informática para la manipulación de cadenas alfanuméricas; no se trata exactamente de un sistema completo de ordenación, pero sí consigue que el tiempo empleado para localizar la información que estamos buscando se reduzca drásticamente.

En realidad, el asunto es muy simple: si nosotros observamos una palabra cualquiera, veremos que la suma de los códigos ASCII de todas sus letras casi, casi, la caracteriza.

Decimos casi porque dos palabras diferente pueden proporcionar el mismo valor numérico.

Para hacer frente a esta posibilidad, mantenemos dos matrices en uso: una de ellas, orden$, almacena las cadenas según van llegando, al azar.

Para que los dedos no realicen el trabajo duro, M.H. AMSTRAD lo hace por ti. Todos los listados que incluyan este icognito se encuentran a tu disposición en un cassette - visual, solicitando.

La segunda, hashtab (100,5), atesora en su dimensión horizontal las palabras que tienen el mismo valor numérico en orden$.

Por ejemplo, si la palabra hola *(valor hash de 20)*, aparece en el tercer elemento de orden$, entonces hashtab*(20,0)* contendrá un tres, de forma que siempre podremos acceder a la palabra *«hola»* mediante la sentencia PRINT orden$ (hashtab(20,0)).

Si posteriormente, en el elemento 50 de orden$ apareciera otra palabra de valor hash 20, se almacenaría en hashtab(20,1), y así sucesivamente.

Por tanto, la utilidad del programa estriba en que PREPARA a una serie de elementos alfanuméricos de forma que pueden ser fácilmente localizados por grupos, y, una vez hallado el grupo en cuestión, investigados mediante cualquier algoritmo de búsqueda, como la búsqueda binaria, para encontrar lo que queremos.

Esto es muy efectivo, porque puede suceder muy fácilmente que de 1.000 palabras, por ejemplo, tengamos que hacer sólo 5 búsquedas o menos.

El programa, más o menos línea por línea, hace lo siguiente:

20 Preparación de la pantalla.

40 Inicialización de las variables globales.

50 Bucle WHILE...WEND de toda la vida.

60 Entrada de datos.

70 Comprobación de si la matriz se ha «acabado».

80 Guardar el nombre en la matriz.

90 Llamada a la rutina de cálculo del valor hash.

100 Colocar en el lugar adecuado.

110 Si ha habido un error, la rutina de la línea 240 en adelante devuelve el valor —1.

170 Inicializar hash a 0.

180-200 Cálculo del valor hash sumando los ASCII del nombre.

210 Para que el valor final quepa dentro de los límites de la matriz, realizamos la operación módulo con el valor 100.

260-300 Encuentra el primer valor no cero dentro de la dimensión horizontal de la matriz marcado por el valor de hash.

310 Coloca el valor.

AMSTRAD CPC - 464

AMSTRAD

ORDENADORI

Esta es la familia de ordenadores personales AMSTRAD. Una familia completa en la que se incluye desde el equipo básico de introducción a la informática hasta el orientado a aplicaciones profesionales. Todo con la filosofía de diseño AMSTRAD que ofrece ordenadores compactos, listos para funcionar sin cableados engorrosos ni necesidad de adquirir periféricos **-con un solo cable a la red-** e incluyendo paquetes de programas de obsequio.

Todos con una tecnología contrastada y fiable basada en el microprocesador Z 80 A, en el Sistema Operativo CP/M - el más extendido para ordenadores de 8 bits- y en una electrónica depurada y con un riguroso control de calidad.

Todos con una extensa biblioteca de programas que se incrementa día a día con títulos para todos los gustos y necesidades.

Todos con una asistencia técnica rápida y eficaz que AMSTRAD ESPAÑA garantiza exclusivamente a los equipos adquiridos a través de su Red Oficial de Distribuidores y acompañados de la Tarjeta de Garantía de AMSTRAD ESPAÑA.

Todos a unos precios increíbles que no admiten comparación con los de cualquier otro ordenador personal de sus características y prestaciones.

AMSTRAD CPC 464.

- Microprocesador Z 80 A• 64K RAM• 32K ROM• Teclado profesional con 32 teclas programables. Sonido estéreo con 3 canales y 8 octavas. Resolución de hasta 640 x 200 puntos. Texto de 20, 40 y 80 columnas. 27 colores. Conectores multiuso, Centronics, joystick etc... Magnetófono incorporado.

TODO POR: 59.000 pts. (monitor verde)
90.000 pts. (monitor color)

EL SUMINISTRO INCLUYE:

- LIBRO "Guia de Referencia del Programador"
- Manual en castellano
- **8 programas de obsequio en cassette** ("Animal, Vegetal y Mineral", "Amsdraw", "Plaga Galáctica", "Fruit Machine", "Admiral Graph Spee", "Amsword", "El Laberinto del Sultán", "OH. Mummy")

PCW - 8256

AMSTRAD CPC - 6128

ES AMSTRAD ¡¡Increíble!!

AMSTRAD CPC 6128.

• Microprocesador Z 80 A• 128 K RAM• 48K ROM (con BASIC Y AMSDOS) • Teclado profesional de 74 teclas (32 programables). Sonido estéreo con 3 canales y 8 octavas. Resolución de hasta 640 x 200 puntos. Texto de 20, 40 y 80 columnas. 27 colores. Conectores multiuso, Centronics, Joystick, etc... Unidad de disco (3", 180K por cara) incorporados.

TODO POR: 99.900 pts.(monitor verde)
127.900 pts.(monitor color)

EL SUMINISTRO INCLUYE:

- Disco con Sistema Operativo CP/M 2.2 y lenguaje DR. LOGO
- Disco con Sistema Operativo CP/M Plus y Utilidades.
- Manual en castellano
- **Disco con 6 programas de obsequio** ("Base de Datos", "Proceso de Textos I", "Ramdom Files","Diseñador de Gráficos", "Puzzle", "Animal, Vegetal y Mineral")

AMSTRAD PCW 8256.

• UNIDAD CENTRAL con microprocesador de Z 80 A, 256K RAM y teclado profesional de 82 teclas (ñ, acento, etc...). PANTALLA DE ALTA RESOLUCION con 90 columnas por 32 líneas de texto. UNIDAD DE DISCO de 3" y 180K por cara. IMPRESORA de tracción/fricción con alineación automática de papel.

TODO POR: 129.900 pts.

EL SUMINISTRO INCLUYE: Procesador de textos LocoScript (en castellano). Sistema operativo CP/M Plus. Mallard BASIC con sistema JETSAM (ficheros indexados). Lenguaje DR. LOGO. Manuales en castellano.

NOTA: Es muy importante verificar la garantía del aparato ya que sólo **AMSTRAD ESPAÑA** puede garantizarle la ordenada reparación y sobre todo materiales de repuesto oficiales (Monitor, ordenador, cassette o unidades de discos).

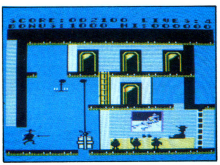
AMSTRAD ESPAÑA

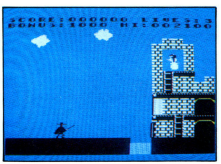
Avda. del Mediterráneo, 9. Tels. 433 45 48 - 433 48 76. 28007 MADRID

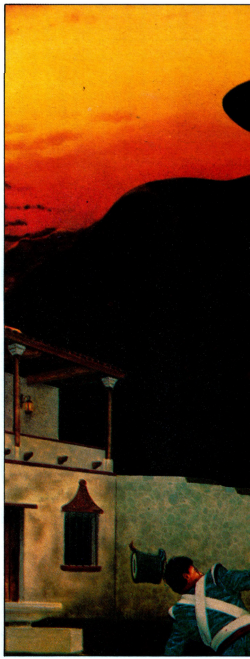
Delegación Cataluña: Tarragona, 110 - Tel. 325 10 58. 08015 BARCELONA

EL ZORRO

La marca del Zorro, signo inequívoco de que algún elemento hostil ha sido eliminado, llega a la pantalla de nuestro ordenador de la mano de U. S. GOLD.







Cucurrucucuuuuuuu.... palomaaaa... cucurrucucuuuuuu... no lloreees... cucurrucucu... cucurrucucuuu.

En las oscuras tinieblas de una noche de mariachis y tequila, una figura se oculta en las sombras. Su marca, la zeta en la frente de sus derrotados rivales. Su ideal: la justicia. Sus armas: el valor y el anonimato.

Los ricos hacendados tiemblan tras los espesos muros de sus mansiones cuando cae la larga noche mejicana. Cada uno reza suplicando que no se produzca la temida visita.

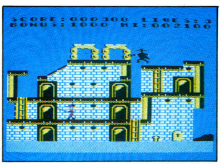
Nadie sabe quién es ni de dónde viene, solamente tenemos vagas descripciones de su aspecto. Vestido de negro de pies a cabeza, antifaz y sombrero de ala ancha ocultan la personalidad, del que los grandes ricachones consideran como el peligro público número uno.

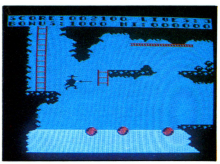
Quién iba a pensar que don Miguelito, el afeminado poseedor de la finca La Tramontana, persona superficial, entretenido en cotilleos de mujeres y obsesionado con vestir a la última moda de Europa, es la figura que vestida de negro amenaza a la alta burguesía.

Diestro espadachín, excelente jinete, ágil escalando muros y caminando por los tejados, es la pesadilla de la guardia presidencial.

Compatible: CPC464, CPC664 y CPC6128

Mr. Joystick





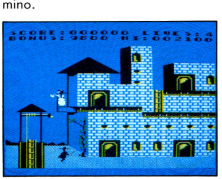
No podemos decir que en esta ocasión los omnipotentes del ALL AMERICAN SOFTWARE, se hayan lucido con su última producción.

Obsesionadas con la idea de lanzar programa tras programa, al precio que cueste, muchas casas de software no cuidan excesivamente la calidad y el acabado de sus programas.

El juego se desarrolla en un escenario de veinte pantallas, cada una de las cuales posee objetos y características especiales que dificultan o facilitan el paso hacia las siguientes.

Caminando por tejados, balcones y deslizándose por tendales de la ropa, camina de pantalla en pantalla en busca de la clave que le permitirá llegar hasta su dama.

María, raptada por el jefe de la guardia, se encuentra encerrada en la torre del fortín presidencial, su rescate no es tarea fácil. Para conseguirlo, el Zorro deberá eliminar cientos de guardias, y encontrar los objetos que le permiten superar los obstáculos que se interponen en su camino.



El movimiento de nuestro personaje, se compone de salto, subir y bajar escalera, caminar, deslizarse por cables y ataques de esgrima y estocada.

El control del mismo, puede realizarse por joystick o teclado.

La resolución del juego puede consumir horas, hasta escalar la torre del cuartel de la guardia presidencial.

La falta de mimo en la realización del programa, tiene su máximo exponente en los gráficos.

Realizados en el modo de menos resolución y más colorido, estos resultan de una terrible simplicidad, colores apagados y falta de sentido artístico.

Nuestro personaje, está realizado con un dibujo caricaturesco y dotado de un movimiento algo brusco, su silueta vaga por la pantalla de lugar en lugar, con indudable seguridad.

El cambio de pantalla a pantalla no es instantáneo, sino que se toma su tiempo, interponiendo una pantalla gris que acentúa dicho lapso.

La excitante aventura del Zorro comienza, en tus manos está aceptar el reto y disfrutar de un programa con gráficos poco trabajados, pero de una emoción indiscutible.

MICROFRUITS

Programa realizado por **José Manuel Rodríguez**

Microfruits es otro intento de tratamiento de la conocida máquina tragaperras con unas características diferenciales respecto a otros intentos.
Se ha intentado una alta fidelidad con el juego original, tratando de conservar todas las características que ofrece típicamente, sin salirse del Basic y sin que, por ello, pierda interés.

Se compone de dos programas encadenados, pero que en caso de carga sin ejecución se accederían independientemente, lo que también es válido para el listado del mismo. El primero es una rutina de presentación que hace menos árida la espera de carga del programa en sí, y que llamo FRUITS. El segundo es el programa en sí y que llamo SLOTS.

En el programa principal, por otra parte, se pueden distinguir en su ejecución tres partes diferenciadas: Un lazo inicial de instrucciones someras e información sobre las combinaciones premiadas. Una rutina para los tiempos de reposo que presenta en lazo todos los displays disponibles. Y el juego propiamente dicho.

Se pueden apostar pesetas (P), duros (D) o monedas de 25 pesetas (M), calculándose el premio en función de la apuesta. Cada apuesta tiene valor para cada jugada individual, pudiéndose cambiar en la siguiente jugada.

Pulsando 1, 2 ó 3 se puede parar cualquier rodillo en cualquier momento de la jugada, correspondiendo cada uno de estos números a la ventana correspondiente, de izquierda a derecha.

Una rutina de avances permite aleatoriamente un número variable de avances manuales al final de la jugada agraciada, que se ejecutan pulsando 1, 2 ó 3 para avanzar la ventana deseada. Si no se pulsa ninguna tecla, se reducen automáticamente los avances sin modificar el display y, por tanto, la combinación obtenida.

Incorpora una rutina aleatoria de premio especial que produce un número variable de combinaciones de premio máximo consecutivas.

SUBRUTINAS

FRUITS	
20	Inicialización
30-40	Pantalla
50	Encadenamiento
SLOTS	
40	Clear
50-140	Estructuración en subrutinas. Secuencia del programa
150-460	Símbolos
470-530	Inicialización
540-750	Generación de cadenas
760-1150	Pantalla
1160-1270	Ventanas
1280-1430	Estado de espera
1440-1750	Comienzo del juego. Apuesta
1760-2140	Jugada. Retención rodillos. Manejo ventanas
2150-2200	Contabilidad
2210-2300	Azar
2310-2410	Especial
2420-2550	Avances
2560-2830	Resultado jugada realizada. Determinación del premio.
2840-3380	Rutina inicial: Presentación, instrucciones, premios.
3390-3760	Sonidos
3770-4230	Marcador y cuadro de honor

Se puede abandonar en cualquier momento pulsando 'T'. Si no se abandona, se termina la serie de jugadas cuando el crédito inicial de 100 pesetas se agota o se alcanza un crédito de 900 pesetas. Se asigna a cada jugador una puntuación en relación con los premios obtenidos en cada jugada y con el crédito final.

Una rutina de marcador con cuadro de honor habilita el programa como juego de competición. Entra automáticamente al final de cada serie de jugadas y usa la puntuación final obtenida. Se comanda con las teclas del cursor ← y →, para los desplazamientos y COPY para tomar las letras del jugador y para regresar al juego y T para terminar, aunque también se puede usar el joystick.

Admite como entrada mayúsculas o minúsculas indistintamente, así como números de ambos teclados.

Se pueden personalizar los displays modificando los literales de las líneas 690 (''JM'') y 700 (''SJ'').


```
10 REM >>>>>>> micro fruits <<<
<<<<<
20 REM >>>>>maquina tragaperras <<<
<
30 REM **** J.M.rodriguez ** 1985 *
****
40 CLEAR:champ=0
50 SYMBOL AFTER 220
60 GOSUB 160
70 GOSUB 480
80 GOSUB 2850
90 GOSUB 770
100 GOSUB 1170
110 GOSUB 1290
120 GOSUB 1450
130 GOTO 90
140 END
150 REM >>>>>> simbolos <<<<<<<<<
160 SYMBOL 220,0,0,0,0,0,1,3,3
170 SYMBOL 221,0,48,64,64,128,128,1
92,192
180 SYMBOL 222,7,7,7,6,6,14,14,13
190 SYMBOL 223,224,224,224,224,224,
240,240,240
200 SYMBOL 224,29,29,61,62,63,31,14
,0
210 SYMBOL 225,248,248,252,60,252,2
48,240,0
220 SYMBOL 226,0,1,3,3,7,7,7,5
230 SYMBOL 227,0,128,192,192,224,22
4,224,224
240 SYMBOL 228,5,5,5,5,5,5,13,9
250 SYMBOL 229,224,224,224,224,224,
224,240,240
260 SYMBOL 230,11,19,23,56,51,31,0
270 SYMBOL 231,240,248,248,28,196,2
48,192,0
280 SYMBOL 232,0,0,0,0,0,3,7,15
290 SYMBOL 233,0,8,16,16,32,224,32,
144
300 SYMBOL 234,31,31,31,59,55,55,59
,59
310 SYMBOL 235,200,248,248,252,252,
252,252,252
320 SYMBOL 236,27,25,28,14,15,7,3,0

330 SYMBOL 237,248,248,248,240,240,
224,192,0
340 SYMBOL 238,0,0,0,1,15,31,55,47
350 SYMBOL 239,0,96,128,0,112,248,2
52,252
360 SYMBOL 240,55,43,55,43,23,27,21
,27
370 SYMBOL 241,252,252,252,252,248,
248,248,248
380 SYMBOL 242,13,15,6,3,3,1,0,0
390 SYMBOL 243,248,240,240,224,192,
128,0,0
400 SYMBOL 244,0,0,0,0,0,1,1,2
410 SYMBOL 245,0,0,28,124,254,254,2
54,254
420 SYMBOL 246,5,5,11,11,23,23,23,4
7
430 SYMBOL 247,254,252,252,252,248,
248,240,240
440 SYMBOL 248,47,47,51,63,30,12,0,
0
450 SYMBOL 249,224,192,128,0,0,0,0,
0
460 RETURN
470 REM >>>>>>> inicializacion <<<<
<
480 MODE 0:PAPER 0:BORDER 5
490 INK 0,1:INK 1,18:INK 2,10:INK 3
,15
500 INK 4,16:INK 5,12:INK 6,25:INK
7,14
510 INK 8,3:INK 9,5:INK 10,0:INK 11
,1,24
520 INK 12,1,19:INK 13,13
530 a=1:punt=0
540 FOR i=1 TO 5
550 n=(i-1)*6
560 a$(i)=CHR$(220+n)+CHR$(221+n)+C
HR$(222+n)+CHR$(223+n)+CHR$(224+n)+
CHR$(225+n)
570 NEXT i
580 a$(6)=CHR$(131)+CHR$(131)+"JM"+
CHR$(140)+CHR$(140)
590 a$(7)=CHR$(216)+CHR$(216)+"SJ"+
CHR$(218)+CHR$(218)
600 b$(1)=LEFT$(a$(1),2)+LEFT$(a$(7
),2)+LEFT$(a$(1),4)+MID$(a$(7),3,2)
+MID$(a$(1),3)+RIGHT$(a$(7),2)+RIGH
T$(a$(1),2)
610 b$(2)=LEFT$(a$(2),2)+LEFT$(a$(2
),2)+LEFT$(a$(2),4)+MID$(a$(2),3,2)
+MID$(a$(2),3)+RIGHT$(a$(2),2)+RIGH
T$(a$(2),2)
620 b$(3)=LEFT$(a$(3),2)+LEFT$(a$(6
),2)+LEFT$(a$(3),4)+MID$(a$(6),3,2)
+MID$(a$(3),3)+RIGHT$(a$(6),2)+RIGH
T$(a$(3),2)
630 b$(4)=LEFT$(a$(4),2)+LEFT$(a$(7
),2)+LEFT$(a$(4),4)+MID$(a$(7),3,2)
+MID$(a$(4),3)+RIGHT$(a$(7),2)+RIGH
T$(a$(4),2)
640 b$(5)=LEFT$(a$(5),2)+LEFT$(a$(6
),2)+LEFT$(a$(5),4)+MID$(a$(6),3,2)
+MID$(a$(5),3)+RIGHT$(a$(6),2)+RIGH
T$(a$(5),2)
650 b$(i)=LEFT$(a$(i),2)+LEFT$(a$(i
),2)+LEFT$(a$(i),4)+MID$(a$(i),3,2)
+MID$(a$(i),3)+RIGHT$(a$(i),2)+RIGH
T$(a$(i),2)
660 FOR i=1 TO 5
670 c$(i)=LEFT$(a$(i),2)+LEFT$(a$(i
),2)+"  "+MID$(a$(i),3,2)+MID$(a$(i
),3,2)+"--"+RIGHT$(a$(i),2)+RIGHT$(
a$(i),2)+"  "
680 NEXT i
690 FOR i=1 TO 5
700 d$(i)="  "+LEFT$(a$(i),2)+LEFT$
(a$(i),2)+"  "+MID$(a$(i),3,2)+MID$
(a$(i),3,2)+"  "+RIGHT$(a$(i),2)+RI
GHT$(a$(i),2)
710 NEXT i
720 e$(1)=LEFT$(a$(4),2)+"    "+MID
$(a$(4),3,2)+"----"+RIGHT$(a$(4),2)
+"    "
730 e$(2)="  "+LEFT$(a$(4),2)+"  --
"+MID$(a$(4),3,2)+"--    "+RIGHT$(a$(
4),2)
740 e$(3)="    "+LEFT$(a$(4),2)+"--
---"+MID$(a$(4),3,2)+"    "+RIGHT$(a
$(4),2)
750 RETURN
760 REM >>>>>>>> pantalla <<<<<<<<
770 MODE 0:CLS
780 WINDOW #1,4,7,9,13:PAPER #1,6
790 WINDOW #2,9,12,9,13:PAPER #2,6
800 WINDOW #3,14,17,9,13:PAPER #3,6
810 PRINT #1,SPC(20);PRINT #2,SPC(
20);:PRINT #3,SPC(20):
820 b=111
830 FOR i=1 TO 3
840 PLOT b,265
850 FOR j=0 TO 20 STEP 2
860 PLOTR -1,1
870 DRAWR 0,-70-j,3
880 DRAWR 95+j,0,3
890 DRAWR 0,70+j,2
900 DRAWR -95-j,0,2
910 NEXT j
920 b=b+160
930 NEXT i
940 WINDOW #6,4,8,19,21
950 WINDOW #7,13,16,19,21
960 PRINT #6,SPC(15);:PRINT #7,SPC(
12)
970 PLOT 92,99
980 FOR j=0 TO 16 STEP 2
990 PLOTR -1,1
1000 DRAWR 0,-25-j,3:DRAWR 170+j,0,
3
1010 DRAWR 0,25+j,2:DRAWR -170-j,0,
2
```

```
1020 NEXT j
1030 PLOT 380,99
1040 FOR j=0 TO 16 STEP 2
1050 PLOTR -1,1
1060 DRAWR 0,-25-j,3:DRAWR 138+j,0,
3
1070 DRAWR 0,25+j,2:DRAWR -138-j,0,
2
1080 NEXT j
1090 LOCATE 3,3:PRINT "M I C R O"
1100 LOCATE 7,5:PRINT "F R U I T S"
```


```
1110 LOCATE 3,18:PRINT "Credito"
1120 LOCATE 12,18:PRINT "Premio"
1130 LOCATE 1,23:PRINT "Inserta mon
eda
      ";
1140 LOCATE 1,25:PRINT " ";CHR$(24)
;"P";CHR$(24);"eseta ";CHR$(24);"D"
;CHR$(24);"uro ";CHR$(24);"M";CHR$(
24);"oneda ";
1150 RETURN
1160 REM >>>> ventanas <<<<<
1170 WINDOW #1,5,6,10,12
1180 WINDOW #1,5,6,10,12
1190 WINDOW #2,10,11,10,12
1200 WINDOW #3,15,16,10,12
1210 WINDOW #4,3,9,16,16
1220 WINDOW #5,12,18,16,16
1230 WINDOW #6,4,8,20,20
1240 PEN #6,10:PAPER #6,1
1250 WINDOW #7,13,16,20,20
1260 PEN #7,10:PAPER #7,1
1270 RETURN
1280 REM >>>>>>> estado de espera <
<<<<
1290 CLS #7:credit=100:PRINT #6,cre
dit;
1300 r=1:GOSUB 1770
1310 FOR i=1 TO 70
1320 e$=INKEY$:e$=UPPER$(e$)
1330 IF e$="T" THEN RETURN
1340 IF e$="P" OR e$="D" OR e$="M"
THEN RETURN
1350 NEXT i
1360 GOTO 1300
1370 d=3:e=0:f=0
1380 FOR i=1 TO 500
1390 e$=INKEY$:e$=UPPER$(e$)
1400 IF e$="T" THEN RETURN
1410 IF e$="P" OR e$="D" OR e$="M"
THEN RETURN
1420 NEXT i
1430 GOTO 1300
1440 REM >>>> comienzo juego <<<<<<
<<
1450 avance=0:special=0:punt=0
1460 IF e$="T" THEN GOTO 1720
1470 IF e$="P" THEN bet=1:GOTO 1490
1480 IF e$="D" THEN bet=5 ELSE bet=
25
1490 IF credit<bet THEN GOTO 2160
1500 credit=credit-bet
1510 IF credit<100 THEN PRINT #6,"
";
1520 PRINT #6,credit;
1530 GOSUB 2220
1540 e=0:f=0
1550 IF special=1 THEN GOSUB 2320
1560 GOSUB 1770
1570 c(1)=a-1:IF c(1)=0 THEN c(1)=5
1580 IF e=2 AND f=3 THEN GOTO 1680
1590 IF e=2 THEN GOTO 1640
1600 IF f=3 THEN d=2:GOSUB 2050:GOT
O 1670
1610 GOSUB 1910
1620 c(2)=a-1:IF c(2)=0 THEN c(2)=5
1630 IF f=3 THEN GOTO 1680
1640 d=3:GOSUB 2050
1650 c(3)=a-1:IF c(3)=0 THEN c(3)=5
1660 GOTO 1680
1670 c(2)=a-1:IF c(2)=0 THEN c(2)=5
1680 IF avance>0 THEN GOSUB 2430
1690 GOSUB 2570
1700 IF credit<1 OR credit>900 THEN
 GOTO 1720
1710 GOSUB 1370:CLS #7:GOTO 1460
1720 IF credit>100 THEN punt=punt+1
NT(credit/500*punt)
1730 GOSUB 3780
1740 GOSUB 480
1750 GOTO 900
1760 REM >>>>> jugada <<<<<<<<<
1770 FOR i=1 TO r
1780 FOR j=1 TO 3
1790 IF i=j+f THEN GOTO 1820
1800 PEN #j,a:IF a=6 OR a=7 THEN PE
N #j,0
1810 PRINT #j,a$(a);
1820 NEXT j
1830 f$=INKEY$
1840 IF f$="1" THEN a=a+1:GOSUB 365
0:RETURN
1850 IF f$="2" AND e<>2 THEN e=2:c(
2)=a:GOSUB 3650
1860 IF f$="3" AND f<>3 THEN f=3:c(
3)=a:GOSUB 3650
1870 a=a+1:IF a=8 THEN a=1
1880 NEXT i
1890 GOSUB 3650
1900 RETURN
1910 FOR i=1 TO s
1920 FOR j=2 TO 3
1930 IF j=f THEN GOTO 2010
1940 PEN #j,a:IF a=6 OR a=7 THEN PE
N #j,0
1950 PRINT #j,a$(a);
1960 NEXT j
1970 f$=INKEY$
1980 IF f$="2" THEN a=a+1:GOSUB 365
0:RETURN
1990 IF f$="3" AND f<>3 THEN f=3:c(
3)=a:GOSUB 3650
2000 a=a+1:IF a=8 THEN a=1
2010 FOR k=1 TO 8:NEXT k
2020 NEXT i
2030 GOSUB 3650
2040 RETURN
2050 FOR i=1 TO t
2060 PEN #d,a:IF a=6 OR a=7 THEN PE
N #d,0
2070 PRINT #d,a$(a);
2080 f$=INKEY$
2090 IF f$="3" THEN a=a+1:GOSUB 365
0:RETURN
2100 a=a+1:IF a=8 THEN a=1
2110 FOR k=1 TO 20:NEXT k
2120 NEXT i
2130 GOSUB 3650
2140 RETURN
2150 REM >>>>>> replay <<<<<
2160 LOCATE 1,24:PRINT SPC(40)
2170 LOCATE 1,24:PRINT "Le quedan "
;credit;" pts."
2180 FOR i=1 TO 500:NEXT i
2190 GOSUB 1130
2200 GOTO 110
2210 REM >>>>>>>> azar <<<<<<<<<<<
2220 r=INT(RND(1)*7)+1
2230 r=r+21
2240 s=INT(RND(1)*7)+1
2250 s=s+7
2260 t=INT(RND(1)*7)+1
2270 t=t+7
2280 u=INT(RND(1)*10)+1
2290 IF u<6 THEN avance=INT(s/3)
2300 RETURN
2310 REM >>>>>>> especial <<<<<<
2320 PAPER #5,0:PEN #5,12:PRINT #5,
"SPECIAL";
2330 GOSUB 3720
2340 v=INT(RND(1)*4)+1
2350 IF v<2 THEN RETURN
2360 r=21:s=14:t=10:a=3
2370 w=INT(RND(1)*4)+1
2380 IF w>2 THEN RETURN
2390 special=0
2400 CLS #5
2410 RETURN
2420 REM >>>>> avances <<<<
2430 PAPER #4,0:PEN #4,12:PRINT #4,
"AVANCES";
2440 GOSUB 3680
2450 FOR i=1 TO 700
2460 e$=INKEY$:IF e$="1" OR e$="2"
OR e$="3" THEN GOTO 2480
2470 NEXT i:GOTO 2520
2480 e=VAL(e$)
2490 c(e)=c(e)+1:IF c(e)=8 THEN c(e
)=1
2500 PEN #e,c(e):IF c(e)=6 OR c(e)=
7 THEN PEN #e,0
2510 PRINT #e,a$(c(e));
2520 avance=avance-1
2530 IF avance>0 THEN GOTO 2440
2540 CLS#4
2550 RETURN
2560 REM >>>>>> suerte <<<<<<<
2570 REM
2580 mul=0
2590 IF c(1)=4 OR c(2)=4 OR c(3)=4
THEN mul=1
2600 IF c(1)=5 OR c(3)=5) AND c(2)
=5 THEN mul=2
2610 IF (c(1)=1 OR c(3)=1) AND c(2)
=1 THEN mul=4
2620 IF (c(1)=3 OR c(3)=3) AND c(2)
=3 THEN mul=6
```

VARIABLES PRINCIPALES

a$ ()	Matriz con los literales de los frutos
b$ () c$ ()	Matrices con las configuraciones para premios
d$ () e$ ()	Matrices con las configuraciones para premios
c ()	Matriz para contabilidad del fruto en cada ventana
credit	Crédito
e$	Apuesta realizada *(P, D o M)*
bet	Valor de la apuesta
punt	Marcador *(puntos)*
f$	Ventana en avance o retención
n$	Nombre del jugador
r s t u	Variables de azar
e f	Variables de control de parada de ventanas
mul	Factor de premio
total	Crédito más premio de cada jugada
na$	Cuadro de honor
champ	Puntuación máxima

Serie Oro

```
2630 IF (c(1)=2 OR c(3)=2) AND c(2)
=2 THEN mul=8
2640 IF c(1)=3 AND c(2)=6 AND c(3)=
3 THEN mul=10
2650 IF c(1)=1 AND c(2)=7 AND c(3)=
1 THEN mul=12
2660 IF c(1)=4 AND c(2)=7 AND c(3)=
4 THEN mul=15
2670 IF c(1)=5 AND c(2)=6 AND c(3)=
5 THEN mul=18
2680 IF c(1)=2 AND c(2)=2 AND c(3)=
2 THEN mul=20
2690 IF mul=20 AND u>8 THEN special
=lin+1
2700 IF mul<10 THEN PRINT #7," ";
2710 PRINT #7,mul;
2720 IF mul=0 THEN GOSUB 3550:GOTO
2740
2730 GOSUB 3410
2740 total=credit+(bet*mul)
2750 FOR i=credit TO total STEP bet
2760 credit=i
2770 IF credit<100 THEN PRINT #6,"
";
2780 IF credit<10 THEN PRINT #6," "
;i
2790 PRINT #6,credit;
2800 GOSUB 3600
2810 NEXT i
2820 punt=punt+mul
```


```
2830 RETURN
2840 >>>>> pantalla de premios <<<<
<<
2850 MODE 0
2860 WINDOW #1,1,6,1,24:PAPER #1,0
2870 WINDOW #2,11,16,1,24:PAPER #2,
0
2880 WINDOW #3,8,9,1,24
2890 PAPER #3,4:PEN #3,0
2900 WINDOW #4,18,19,1,24
2910 PAPER #4,4:PEN #4,0
2920 WINDOW #5,1,20,24,24
2930 PAPER #5,4:PEN #5,0
2940 WINDOW #6,2,17,3,5
2950 PAPER #6,4:PEN #6,0
2960 WINDOW #7,6,18,6,8
2970 PAPER #7,4:PEN #7,0
2980 CLS
2990 PRINT #6,"                M I C R
O                 ";
3000 PRINT #7,"                F R U
I T S                 ";
3010 LOCATE 1,13:PRINT "Teclas:"
3020 LOCATE 1,14:PRINT "=======
3030 LOCATE 3,16:PRINT "P-D-M  para
 jugar"
3040 LOCATE 3,18:PRINT "1-2-3  para
 parar"
3050 LOCATE 3,20:PRINT "  T    term
inar"
3060 PRINT #5," Pulsa una tecla
";
3070 GOSUB 3340
3080 IF e$<>"" THEN RETURN
3090 CLS
3100 LOCATE 4,11
3110 PRINT "P R E M I O S"
3120 LOCATE 4,13
3130 PRINT "============"
3140 FOR i=1 TO 15
3150 GOSUB 3600
3160 IF INKEY$<>"" THEN RETURN
3170 NEXT i
3180 CLS
3190 PEN #1,2:PRINT #1,b$(2);
3200 PEN #1,5:PRINT #1,b$(5);
3210 PEN #1,4:PRINT #1,b$(4);
3220 PEN #1,1:PRINT #1,b$(1);
3230 PEN #1,3:PRINT #1,b$(3);
3240 PEN #1,2:PRINT #1,c$(2);d$(2);

3250 PEN #1,3:PRINT #1,c$(3);
3260 PEN #2,3:PRINT #2,d$(3);
3270 PEN #2,1:PRINT #2,c$(1);d$(1);
3280 PEN #2,5:PRINT #2,c$(5);d$(5);
3290 PEN #2,4:PRINT #2,e$(1);e$(2);
e$(3);
3300 PRINT #3," 20    18    15
12   10     8     8     6  ";
3310 PRINT #4," 6     4     4
 2     2     1     1     1  ";
3320 GOSUB 3340
3330 GOTO 2850
3340 FOR i=1 TO 600
3350 e$=INKEY$:e$=UPPER$(e$)
3360 IF e$<>"" THEN RETURN
3370 NEXT i
3380 RETURN
3390 REM >>>>>>>>>>> sonidos <<<<<<
<<<<<<<<<<
3400 REM ..... premios .....
.......
3410 FOR k=mul*10 TO 10 STEP -30
3420 FOR l=k TO 10 STEP -10
3430 SOUND 1,1,10
3440 NEXT l
3450 FOR l=10 TO k STEP 10
3460 SOUND 1,1,10
3470 NEXT l
3480 NEXT k
3490 FOR k= 1 TO 5
3500 SOUND 1,0,15
3510 SOUND 1,50,20
3520 NEXT k
3530 RETURN
3540 REM ..... no premio ...
.......
3550 FOR k=90 TO 800 STEP 50
3560 SOUND 1,k,10
3570 NEXT k
3580 RETURN
3590 REM ..... caida monedas
 .......
3600 FOR k=1 TO 200 STEP 10
3610 SOUND 1,k,1
3620 NEXT k
3630 RETURN
3640 REM ..... parar slots .
.......
3650 SOUND 1,1000,20
3660 RETURN
3670 REM ..... avances .....
.......
3680 SOUND 1,90,10
3690 SOUND 1,0,10
3700 RETURN
3710 REM ..... especial ....
.......
3720 FOR k=1 TO 20
3730 SOUND 1,40,10
3740 SOUND 1,0,10
3750 NEXT k
3760 RETURN
3770 REM  #### rutina MARCADOR ####
#
3780 MODE 1:BORDER 0:CLS:re=0
3790 PRINT:PRINT TAB(4);"Cuadro de
Honor para MICRO-FRUITS"
3800 PRINT:PRINT
3810 FOR a=1 TO 5
3820 PRINT:PRINT TAB(4);a;"....";na
$(a);TAB(30);sco(a)
3830 NEXT a
3840 FOR a =5 TO 1 STEP -1
3850 IF punt>champ(a) AND punt<cham
p(a-1) THEN rea
3860 NEXT
3870 IF re=0 THEN 4210
3880 FOR a=4 TO re STEP -1:na$(a+1)
=na$(a):champ(a+1)=champ(a):NEXT
3890 na$(re)="":champ(re)=sc
3900 CLS:PRINT:PRINT TAB(4);"Cuadro
 de honor para  MICRO-FRUITS"
3910 PRINT:PRINT
3920 FOR a=1 TO 5
3930 PRINT:PRINT TAB(4);a;"....";na
$(a);TAB(30);champ(a)
3940 NEXT a
3950 PRINT:PRINT "For favor, escrib
e tu nombre ":in$="ABCDEFGHIJKLMNOPQ
RSTUVWXYZ.1"
3960 PRINT:PRINT TAB(5);n$;x=5
3970 PRINT:PRINT "              y al t
erminar pon ."
3980 LOCATE x,19:PRINT "-"
3990 LOCATE 10,22:PRINT na$(re)
4000 a$=INKEY$:IF a$="" THEN GOTO 4
000
4010 IF INKEY(74)=0 OR INKEY(8)=0 T
HEN LOCATE x,19:PRINT " ":x=x-1
4020 IF INKEY(75)=0 OR INKEY(1)=0 T
HEN LOCATE x,19:PRINT " ":x=x+1
4030 IF (INKEY(77)=0 OR INKEY(9)=0)
 AND MID$(n$,(x-4),1)=" " THEN 4070

4040 IF INKEY(77)=0 OR INKEY(9)=0 T
HEN na$(re)=na$(re)+MID$(n$,(x-4),1
)
4050 IF LEN(na$(re))=15 THEN 4070
4060 GOTO 3980
4070 MODE 1:CLS
4080 PRINT:PRINT TAB(5);"Cuadro de
Honor para MICRO-FRUITS"
4090 PRINT:PRINT
4100 FOR a=1 TO 5
4110 PRINT:PRINT TAB(4);a;"....";na
$(a);TAB(30);champ(a)
4120 NEXT
4130 LOCATE 2,20:PRINT "Pulsa 'disp
aro' para jugar otra vez"
4140 LOCATE 2,22:PRINT "Pulsa 'T' p
ara terminar"
4150 a$=INKEY$:IF a$="" GOTO 4150
4160 IF a$="t" OR a$="T" GOTO 4230
4170 a$=""
4180 IF INKEY(77)=0 OR INKEY(9)=0 T
HEN 4200
4190 GOTO 4180
4200 GOSUB 480:RETURN
4210 IF punt<champ(1) THEN 4070
4220 re=1:GOTO 3880
4230 MODE 2:PEN 1:END
```

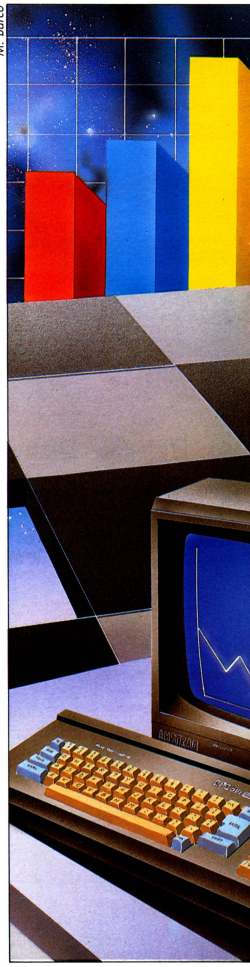
Para que tus dedos no realicen el trabajo duro, M.H. AMSTRAD lo hace por ti. Todos los listados que incluyan este logotipo se encuentran a tu disposicion en un cassette mensual, solicitanolo.

WINDOWKEY

VENTANA INFORMATIVA PARA TECLAS DE FUNCION

Por Alberto Súñer

Todos hemos utilizado en más de una ocasión las posibilidades que posee el Amstrad para definir nuestras propias teclas de función, con lo que podemos efectuar mucho más cómodamente la labor de programación.


M. Barco

Aunque la posibilidad de utilizar dichas teclas posee unas ventajas enormes, en muchas ocasiones nos encontramos con el inconveniente de no acordarnos en qué tecla tenemos definido un comando concreto, por lo que tenemos que ir comprobando cada una de las teclas hasta encontrar la palabra clave que en ese momento deseamos utilizar.

Hemos pensado que una de las formas más sencillas de resolver este problema, sería crear una ventana informativa en la parte inferior de la pantalla, que nos presentará en cada momento cada una de las teclas de función definidas por nosotros, para así no tener que acordarnos de qué comando se encuentra en cada una de dichas teclas.

Pues bien, en el programa que acompaña a este artículo se encuentra la solución. Dicha rutina, se encarga de mostrarnos en cada momento todas las teclas de función, en una ventana situada en la parte baja de la pantalla. De este modo, podremos visualizar cuáles son las nuevas funciones de cada una de las teclas.

Para la utilización de esta ayuda de programación, únicamente deberéis copiar el listado para aquellos que posean ensamblador o bien el programa cargador en Basic para los que no tengan en su poder un ensamblador.

Una vez copiado el listado ejecutaremos el programa Basic. Si hemos cometido algún error, el programa nos lo indicará por lo que deberemos corregir las datas de nuestro programa.

En caso contrario podemos salvar ese programa en cinta o disco, de la siguiente forma:

SAVE «**TECLAS**», B, &A000, &150

Para la ejecución de la rutina una vez en memoria, únicamente deberemos teclear lo siguiente:

CALL &A000

A partir de este momento dispondremos de dos nuevos comandos que podrán ser utilizados desde Basic, gracias a las posibilidades de los RSX. Estos nuevos comandos son los descritos a continuación:

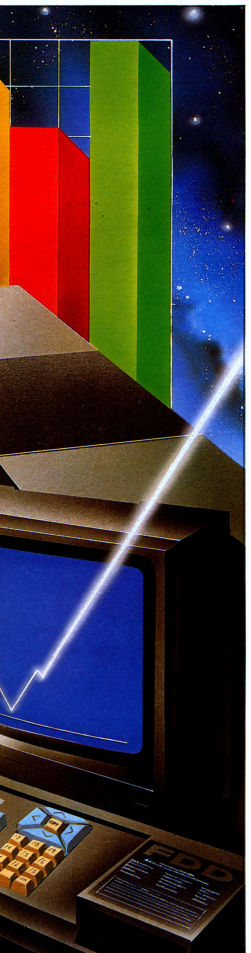
|ON... Activa la ventana informativa en la parte inferior de la pantalla.

|OFF... Desactiva dicha ventana informativa.

Como ya habéis podido observar, el programa utiliza las posibilidades de los RSX, para una mayor comodidad del usuario y además utiliza las facilidades que ofrece el **Amstrad** para el tratamiento de interrupciones.

Nos podríamos preguntar cuál es la necesidad de las interrupciones en el programa que estamos describiendo. Pues bien, la respuesta sería que debido a los posibles cambios en las teclas de función, la ventana informativa quedaría obsoleta si no se produjeran las modificaciones pertinentes y para que estas modificaciones se produzcan, no nos queda otro remedio que utilizar las interrupciones. Así pues, la rutina se ejecuta cada 1/50 segundos y se producen las modificaciones si éstas son precisas.

Vamos a entrar en la descripción del programa propiamente dicho. En primer lugar podemos observar que las primeras líneas del programa son suficiente conocidas por nosotros, ya que su misión es indicar al sistema que existen comandos RSX. A partir de ahí, definimos los nuevos comandos y su tabla de saltos, para que el sistema sepa a qué atenerse cuando un comando RSX es llamado desde Basic.



El programa en sí, empieza en la línea 220 que es donde se realiza el tratamiento de interrupciones, indicándole al sistema que se va a producir una interrupción y la posición de memoria a donde tiene que saltar cuando ésta se produzca. En la línea 300 podemos ver la rutina encargada de desactivar la interrupción, cuando no deseamos ver en pantalla la ventana informativa.

Una vez realizados estos pasos preliminares, entramos en la rutina a la que saltará el ordenador cuando se produzca la interrupción. En primer lugar tenemos la definición de ventanas, una para la información, en la parte inferior y otra en la parte superior, en la cual nosotros podremos realizar nuestros programas o ejecutarlos.

A continuación se produce un chequeo que comprueba en qué modo de pantalla se está trabajando, si en ese momento se trabaja en modo 0, no se ejecuta la rutina, ya que debido a la anchura de caracteres, no se podrían imprimir en pantalla el contenido de las teclas de función. Si está en otro modo, entonces se hacen los cambios necesarios para que la presentación sea la correcta en cada modo.

Ahora nos encontramos con un bucle, encargado de comprobar si se ha producido algún cambio en el contenido de las teclas de función, o sea, si nosotros hemos cambiado alguna de esas teclas, si es así, se salta a la rutina encargada de imprimir en pantalla el contenido de dichas teclas. Si al salir de dicho bucle se comprueba que no se ha producido ningún cambio, entonces se retorna al Basic.

Por último nos encontramos con la rutina encargada de la impresión de las teclas de función en la ventana informativa. Al inicio de esta rutina se carga el registro doble DE con la posición de memoria en donde se encuentra almacenado el contenido de las teclas de función.

El primer Byte de esa posición de memoria, nos indicará la longitud de la cadena de la primera tecla de función y a continuación se encuentra dicha cadena almacenada en forma de caracteres ASCI, el byte que sigue a continuación nos indicará la longitud de la cadena de caracteres correspondientes a la tecla de función siguiente, y a continuación tendremos los caracteres correspondientes a esa segunda tecla de función, y así sucesivamente hasta llegar a la cadena de caracteres que definen la última tecla.

Program Acción

Por lo tanto, lo que se hace para imprimir el contenido de cada una de las teclas, es tomar en primer lugar su longitud y la traspasamos al registro B, que será el encargado de indicar cuántas veces se debe repetir el bucle de impresión, a continuación tomamos el valor del siguiente byte que pertenecerá ya a la cadena que define la primera tecla, y lo imprimimos llamando a la rutina del firmware encargada de ello, y así hasta completar el bucle.

Una vez hecho esto se vuelve a repetir el mismo proceso para la impresión del contenido de la segunda tecla de función, repitiendo el proceso hasta llegar a la última tecla de función almacenada en esa posición de memoria.

Por último, diremos que la rutina que se encuentra listada a continuación, funcionará perfectamente en un **Amstrad** modelo 464, los usuarios que posean otro modelo, deberán tener en cuenta la siguiente modificación:

La etiqueta con el nombre de:

TECDEF: EQU #B446

deberá cambiarse por la siguiente:

TECDEF: EQU #B500

ésta se encuentra situada en la línea 1390 del listado ensamblador.

Para aquellos que prefieran el cargador Basic, deberán cambiar las líneas 300 y 400 por las que se indican a continuación:

300 DATA 33, 144, 181, 221, 126, 0, 71

400 DATA 48, 161, 17, 144, 181, 213, 221

además la línea 50 deberá ser la siguiente:

50 IF SUMA < > &75FF THEN PRINT «ERROR EN DATAS»

Una vez efectuados dichos cambios el programa correrá en un **Amstrad** 664 y en un 6128.

Nota: *Las teclas de función están numeradas del 0 al 9. Para definir una, se usa la orden KEY, seguida del número de tecla y de la cadena de órdenes. Por ejemplo:*

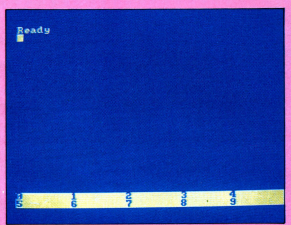
KEY 0, «LIST»+CHR$(13)

al pulsar CTRL+0, aparecerá en pantalla el listado del programa.

PROGRAMA I

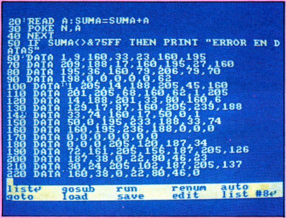
```
10 FOR N=&A000 TO &A14A
20 READ A:SUMA=SUMA+A
30 POKE N,A
40 NEXT
50 IF SUMA<>&75FF THEN PRINT "ERROR
 EN DATAS"
60 DATA 1,9,160,33,23,160,195
70 DATA 209,188,17,160,195,27,160
80 DATA 195,36,160,79,206,79,70
90 DATA 198,0,0,0,0,0,62
100 DATA 1,205,14,188,205,45,160
110 DATA 201,205,68,160,62,1,205
120 DATA 14,188,201,33,80,160,6
130 DATA 129,17,87,160,205,239,188
140 DATA 33,74,160,17,50,0,1
150 DATA 50,0,195,233,188,33,74
160 DATA 160,195,236,188,0,0,0
170 DATA 0,0,0,0,0,0,0
180 DATA 0,0,0,205,120,187,34
190 DATA 72,161,205,156,187,205,126
200 DATA 187,38,0,22,80,46,23
210 DATA 30,24,205,102,187,205,137
220 DATA 160,38,0,22,80,46,0
230 DATA 30,22,205,102,187,42,72
240 DATA 161,205,117,187,205,123,18
7
250 DATA 205,156,187,201,205,17,188
260 DATA 50,71,161,254,0,200,33
270 DATA 1,1,205,117,187,205,96
280 DATA 187,254,143,40,56,62,0
290 DATA 50,68,161,33,0,0,221
300 DATA 33,70,180,221,126,0,71
310 DATA 221,35,221,126,0,221,35
320 DATA 22,0,95,25,16,245,58
330 DATA 68,161,60,50,68,161,254
340 DATA 10,40,2,24,226,237,91
350 DATA 69,161,34,69,161,122,188
360 DATA 32,3,123,189,200,205,108
370 DATA 187,62,0,50,68,161,58
380 DATA 71,161,254,1,32,6,221
390 DATA 33,28,161,24,4,221,33
400 DATA 48,161,17,70,180,213,221
410 DATA 102,0,221,35,221,110,0
420 DATA 221,35,205,117,187,209,26
430 DATA 71,19,197,26,213,205,93
440 DATA 187,209,19,193,16,245,58
450 DATA 68,161,60,50,68,161,254
460 DATA 10,200,24,215,1,1,9
470 DATA 1,17,1,25,1,32,1
480 DATA 1,2,9,2,17,2,25
490 DATA 2,32,2,1,1,17,1
500 DATA 33,1,49,1,64,1,1
510 DATA 2,17,2,33,2,49,2
520 DATA 64,2,0,0,0,1,0
530 DATA 0,0,0,0,0,0,0
```

```
A000                    10          ORG   #A000
A000 0189A0             20          LD    BC,TABLA
A003 2117A0             30          LD    HL,ESPACE
A006 C301BC             40          JP    RSX
A009 11A0               50 TABLA:   DEFW  NAME
A00B C31BA0             60          JP    ON
A00E C324A0             70          JP    OFF
A011 4F                 80 NAME:    DEFM  "O"
A012 CE                 90          DEFB  "N"+#80
A013 4F46               100         DEFM  "OF"
A015 C6                 110         DEFB  "F"+#80
A016 00                 120         DEFB  0
A017                    130 ESPACE: DEFS  4
A01B 3E01               140 ON:     LD    A,1
A01D CD0EBC             150         CALL  SETMOD
A020 CD2DA0             160         CALL  ACTI
A023 C9                 170         RET
A024 CD44A0             180 OFF:    CALL  DESAC
A027 3E81               190         LD    A,1
A029 CD0EBC             200         CALL  SETMOD
A02C C9                 210         RET
A02D 2158A0             220 ACTI:   LD    HL,BLOCK
A030 0681               230         LD    B,129
A032 1157A0             240         LD    DE,EVENT
A035 CDEFBC             250         CALL  INICIA
A038 2144A0             260         LD    HL,RELOJ
A03B 113200             270         LD    DE,50
A03E 013200             280         LD    BC,50
A041 C3E9BC             290         JP    TICKSI
A044 2144A0             300 DESAC:  LD    HL,RELOJ
```




```
A047 C3ECBC             310         JP    TICKNO
A04A                    320 RELOJ:  DEFS  6
A050                    330 BLOCK:  DEFS  7
A057 CD78BB             340 EVENT:  CALL  GETCUR
A05A 2248A1             350         LD    (CURSOR),HL
A05D CD9CBB             360         CALL  TXTINV
A060 CD7EBB             370         CALL  CUROFF
A063 2600               380         LD    H,0
A065 1650               390         LD    D,80
A067 2E17               400         LD    L,23
A069 1E18               410         LD    E,24
A06B CD66BB             420         CALL  WINDOW
A06E CD89A0             430         CALL  ENTRA
A071 2600               440         LD    H,0
A073 1650               450         LD    D,80
A075 2E00               460         LD    L,0
A077 1E16               470         LD    E,22
A079 CD66BB             480         CALL  WINDOW
A07C 2A48A1             490         LD    HL,(CURSOR)
A07F CD75BB             500         CALL  LOCATE
A082 CD7EBB             510         CALL  CURON
A085 CD9CBB             520         CALL  TXTINV
A088 C9                 530         RET
A089 CD11BC             540 ENTRA:  CALL  GETMOD
A08C 3247A1             550         LD    (MODO),A
A08F FE00               560         CP    0
A091 C8                 570         RET   Z
A092 210101             580         LD    HL,#0101
A095 CD75BB             590         CALL  LOCATE
A098 CD60BB             600         CALL  READ
A09B FE8F               610         CP    143
```

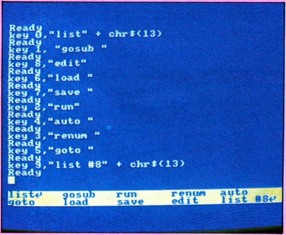
```
A09D  2838       620           JR    Z,EMP
A09F  3E00       630           LD    A,0
A0A1  3244A1     640           LD    (FLAG),A
A0A4  210000     650           LD    HL,0
A0A7  DD2146B4   660           LD    IX,TECDEF
A0AB  DD7E00     670  BUS:     LD    A,(IX)
A0AE  47         680           LD    B,A
A0AF  DD23       690           INC   IX
A0B1  DD7E00     700  BUS1:    LD    A,(IX)
A0B4  DD23       710           INC   IX
A0B6  1600       720           LD    D,0
A0B8  5F         730           LD    E,A
A0B9  19         740           ADD   HL,DE
A0BA  10F5       750           DJNZ  BUS1
A0BC  3A44A1     760           LD    A,(FLAG)
A0BF  3C         770           INC   A
A0C0  3244A1     780           LD    (FLAG),A
A0C3  FE0A       790           CP    10
A0C5  2802       800           JR    Z,SIG
A0C7  18E2       810           JR    BUS
A0C9  ED5B45A1   820  SIG:     LD    DE,(SUMA)
A0CD  2245A1     830           LD    (SUMA),HL
A0D0  7A         840           LD    A,D
A0D1  BC         850           CP    H
A0D2  2003       860           JR    NZ,EMP
```




```
A0D4  7B         870           LD    A,E
A0D5  BD         880           CP    L
A0D6  C8         890           RET   Z
A0D7  CD6CBB     900  EMP:     CALL  CLEAR
A0DA  3E00       910           LD    A,0
A0DC  3244A1     920           LD    (FLAG),A
A0DF  3A47A1     930           LD    A,(MODO)
A0E2  FE01       940           CP    1
A0E4  2006       950           JR    NZ,MOD
A0E6  DD211CA1   960           LD    IX,POS
A0EA  1804       970           JR    YASTA
A0EC  DD2130A1   980  MOD:     LD    IX,POS1
A0F0  1146B4     990  YASTA:   LD    DE,TECDEF
A0F3  D5        1000  BUC1:    PUSH  DE
A0F4  DD6600    1010           LD    H,(IX)
A0F7  DD23      1020           INC   IX
A0F9  DD6E00    1030           LD    L,(IX)
A0FC  DD23      1040           INC   IX
A0FE  CD75BB    1050           CALL  LOCATE
A101  D1        1060           POP   DE
A102  1A        1070           LD    A,(DE)
A103  47        1080           LD    B,A
A104  13        1090           INC   DE
A105  C5        1100  BUC:     PUSH  BC
A106  1A        1110           LD    A,(DE)
A107  D5        1120           PUSH  DE
A108  CD5DBB    1130           CALL  PRINT
A10B  D1        1140           POP   DE
A10C  13        1150           INC   DE
A10D  C1        1160           POP   BC
A10E  10F5      1170           DJNZ  BUC
A110  3A44A1    1180           LD    A,(FLAG)
A113  3C        1190           INC   A
A114  3244A1    1200           LD    (FLAG),A
A117  FE0A      1210           CP    10
```

ProgramAcción

```
A119  C8        1220           RET   Z
A11A  18D7      1230           JR    BUC1
BC0E            1240  SETMOD:  EQU   #BC0E
BCD1            1250  RSX:     EQU   #BCD1
BCEF            1260  INICIA:  EQU   #BCEF
BCE9            1270  TICKSI:  EQU   #BCE9
BCEC            1280  TICKNO:  EQU   #BCEC
BB78            1290  GETCUR:  EQU   #BB78
BB9C            1300  TXTINV:  EQU   #BB9C
BB7E            1310  CUROFF:  EQU   #BB7E
BB66            1320  WINDOW:  EQU   #BB66
BB75            1330  LOCATE:  EQU   #BB75
BB7B            1340  CURON:   EQU   #BB7B
BC11            1350  GETMOD:  EQU   #BC11
BB60            1360  READ:    EQU   #BB60
BB6C            1370  CLEAR:   EQU   #BB6C
BB5D            1380  PRINT:   EQU   #BB5D
B446            1390  TECDEF:  EQU   #B446
A11C  01010901  1400  POS:     DEFB  1,1,9,1,17,1,25,1
A124  20010102  1410           DEFB  32,1,1,2,9,2,17,2
A12C  19022002  1420           DEFB  25,2,32,2
A130  01011101  1430  POS1:    DEFB  1,1,17,1,33,1,49,1
A138  40010102  1440           DEFB  64,1,1,2,17,2,33,2
A140  31024002  1450           DEFB  49,2,64,2
A144            1460  FLAG:    DEFS  1
A145            1470  SUMA:    DEFS  2
A147  01        1480  MODO:    DEFB  1
A148            1490  CURSOR:  DEFS  2
```



ETIQUETAS

ACTI	A02D	BLOCK	A050	BUC	A105
BUC1	A0F3	BUS	A0AB	BUS1	A0B1
CLEAR	BB6C	CUROFF	BB7E	CURON	BB7B
CURSOR	A148	DESAC	A044	EMP	A0D7
ENTRA	A089	ESPACE	A017	EVENT	A057
FLAG	A144	GETCUR	BB78	GETMOD	BC11
INICIA	BCEF	LOCATE	BB75	MOD	A0EC
MODO	A147	NAME	A011	OFF	A024
ON	A01B	POS	A11C	POS1	A130
PRINT	BB5D	READ	BB60	RELOJ	A04A
RSX	BCD1	SETMOD	BC0E	SIG	A0C9
SUMA	A145	TABLA	A009	TECDEF	B446
TICKNO	BCEC	TICKSI	BCE9	TXTINV	BB9C
WINDOW	BB66	YASTA	A0F0		

INSTRUCCIONES DE ROTACION Y DESPLAZAMIENTO (II)

Como recordaréis en el último capítulo hablábamos de una serie de instrucciones de rotación y desplazamiento, que actuaban sobre el acumulador. Pues bien, hoy empezaremos hablando de esas mismas instrucciones pero referidas a diferentes registros.

Las instrucciones de rotación, aplicadas al resto de los registros, se representan de la siguiente forma:

RLCm
RLm
RRCm
RRm

el operando **«m»** puede ser cualquiera de los registros simples B, C, D, E, H, L, A o bien (HL), (IX+d) o IY+d).

Como ya hemos dicho, actúan exactamente igual que las explicadas en el capítulo anterior: la única diferencia reside en el operando «m» que sustituye al registro A.

Dicho esto supongo que ya habréis intuido el resultado de la ejecución de los dos primeros programas que presentamos. El primero constituye una rotación a la derecha del contenido de la posición de memoria a la que apunta el registro doble HL, teniendo en cuenta que el contenido del bit 0 se copia en el bit 7 y también en el flag Carry. El segundo programa, también constituye una rotación a la derecha, esta vez de la dirección de memoria a la que apunta (IX+1), pero esta vez, el contenido del bit 0 se copia en el flag Carry y el contenido de éste se traspasa al bit 7.

Instrucción aritmética SLA

Vamos a ver ahora otra nueva instrucción que produce un desplazamiento en el registro al que se aplica. Se representa de la siguiente manera:

SLA m

el operando 'm' puede ser cualquiera de los registros B, C, D, E, H, L, A o bien (HL), (IX+d) o (IY+d).

La ejecución de esta instrucción produce un desplazamiento aritmético en el contenido del operando 'm'. El bit 0 se pone a cero, el contenido previo del bit 0 se copia en el bit 1, el contenido previo de éste se copia en el bit 2 y así sucesivamente. El contenido del bit 7 se copia en el flag Carry del registro F.

Esquemáticamente produce el siguiente efecto:

C←7←6←5←4←3←2←1←0 ← 0

Tras la ejecución de esta instrucción, el contenido del bit 0 siempre será cero, y el contenido del registro en cuestión se ve multiplicado por dos.

Un ejemplo de actuación lo tenemos en el programa número 3. Aquí hemos aprovechado esta instrucción para multiplicar por dos el contenido del acumulador; así pues, si inicialmente hemos cargado el acumulador con el valor 2, después de la ejecución del programa el contenido del acumulador será 4. Esto puede observarse mirando la posición de memoria #7000 después de la ejecución del programa.

Ahora bien, si nos fijamos en el siguiente programa, veremos que al cargar el acumulador con el valor 128, tras la ejecución de SLA A, el contenido del acumulador vale cero en lugar de multiplicarse por dos. Así pues, podremos aprovechar la capacidad de duplicar el contenido de un registro, siempre y cuando el contenido del bit 7 de éste contenga un cero.

Instrucción SRA

La siguiente instrucción a estudiar se representa de la forma indicada a continuación:

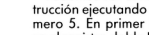
SRA m

donde el operando **'m'** puede ser cualquiera de los registros B, C, H, L, D, E, A o bien (HL), (IX+d) o (IY+d).

La ejecución produce un desplazamiento aritmético hacia la derecha del contenido del operando m. El contenido del bit 7 se copia en el bit 6, el contenido previo del bit 6 se copia en el bit 5 y así sucesivamente.

El contenido del bit 0 se copia en el flag Carry del registro F, y el contenido del bit 7 no cambia.

Esquemáticamente se puede representar como sigue:



Podemos ver cómo actúa esta instrucción ejecutando el programa número 5. En primer lugar cargamos en el registro doble HL la posición de memoria #7000, y luego mediante la instrucción SRA (HL) producimos un desplazamiento hacia la derecha.

Instrucción SRL

Otra instrucción de este tipo es la que se representa a continuación:

SRL m

donde el operando 'm' puede ser como hemos dicho en repetidas ocasiones, cualquier registro o bien (HL), (IX+d) o IY+d).

Tras la ejecución, el comando 'm' se desplaza hacia la derecha. El contenido del bit 7 se copia en el bit 6, el contenido previo de éste se copia en el bit 5 y así sucesivamente. El contenido del bit 0 se copia en el flag Carry y el bit 7 se resetea (se pone a cero), por lo que el contenido del registro queda dividido por dos.

El efecto producido por dicha instrucción podemos representarlo de la siguiente manera:

0→7→6→5→4→3→2→1→0→C

En el programa número 6 podemos ver una posible utilización práctica. El programa en cuestión es capaz de distinguir entre un número par y un número impar, eso sí, siempre en el rango de 0 a 255, que es el máximo valor que se puede cargar en un registro.

La explicación es bien sencilla: dado que el bit 0 es el único bit que puede dar a un registro un valor impar, lo que hacemos es pasar el contenido de ese bit al flag Carry mediante la instrucción SRL A. Una vez hecho esto, si el contenido del Carry es 1, podremos decir que el valor que contenía el registro dado era impar, por el contrario si el Carry contiene un 0, el valor de dicho registro era par.

Instrucciones de manipulación de nibbles: RLD y RRD

Vamos a ver ahora una instrucción algo más complicada que las anteriores, pero muy potente. Se representa como sigue:

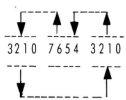
RLD

Esta instrucción actúa sobre el acumulador y sobre el registro doble HL.

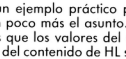
Tras la ejecución el contenido de los cuatro bits de orden bajo, *(bits 3, 2, 1, 0)*, de la dirección de memoria a la que apunta HL se copia en los bits de orden alto, *(bits, 7, 6, 5, 4)*, de esa misma dirección de memoria. El contenido previo de esos cuatro bits de orden alto, se copian en los

cuatro de orden bajo del acumulador, (registro A), y el contenido previo de los cuatro bits de orden bajo del acumulador se copian en los cuatro bits de orden bajo de (HL). El contenido de los cuatro bits de orden alto del acumulador no quedan afectados.

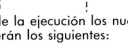
Vamos a ver esquemáticamente el efecto producido tras la ejecución de dicha instrucción:



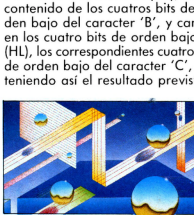
Ahora un ejemplo práctico para aclarar un poco más el asunto. Supongamos que los valores del acumulador y del contenido de HL sean los siguientes.



después de la ejecución los nuevos valores serán los siguientes:



En el programa número 7 podemos ver un ejemplo de aplicación de RLD. En primer lugar, cargamos el acumulador con el código ASCI de la letra 'A', luego apuntamos el registro doble HL a la posición de memoria #7000, cargando en esa posición el valor #23 y luego mediante la ayuda de la instrucción RLD logramos imprimir en pantalla las tres primeras letras del abecedario. **¿Dónde está el misterio?** Bueno, pues la explicación es muy sencilla, sólo basta con mirar en cómo actúa dicha instrucción. Dado que estos tres caracteres tienen los bit de orden bajo exactamente igual, entonces lo que hemos hecho es cargar en los cuatro bits de orden alto de (HL), con el contenido de los cuatros bits de orden bajo del caracter 'B', y cargar en los cuatro bits de orden bajo de (HL), los correspondientes cuatro bits de orden bajo del caracter 'C', obteniendo así el resultado previsto.



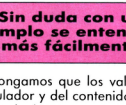
Instrucción RRD

La siguiente instrucción que vamos a ver, y con ella terminamos el capítulo dedicado a instrucciones de rotación y desplazamiento de bits, es muy semejante a la anterior, y se representa como sigue:

RRD

Tras la ejecución, el contenido de los cuatro bits de orden bajo de la posición de memoria a la que apunta HL, se copian en los cuatro bits de menor orden del acumulador. El contenido previo de los cuatro bits de orden bajo del acumulador se copian en los cuatro bits de orden alto de (HL). El contenido previo de los cuatro bits de mayor orden de (HL) se copian en los cuatro bits de orden bajo de (HL). El contenido de los cuatro bits de mayor orden del registro A no son afectados por esta instrucción.

Gráficamente, la operación que realiza RRD es:



Sin duda con un ejemplo se entenderá más fácilmente

Supongamos que los valores del acumulador y del contenido de HL, son los siguientes:

después de la ejecución de RRD, obtendremos los siguientes valores:

En el programa número 8 podemos ver una aplicación práctica. La filosofía es idéntica a la del programa anterior, pero aquí lo que hacemos es imprimir los tres primeros números; de esta forma podemos guardar en una sola posición de memoria los códigos ASCI de dos números, ahorrando una posición de memoria por cada número a imprimir.

PROGRAMAS

```
Hisoft GENA3.1 Assembler. Page      1.
Pass 1 errors: 00
                        10 ;PROGRAMA-1
                        20 ;
A000                    30          ORG  #A000
A000  210070            40          LD   HL,#7000
A003  CB0E              50          RRC  (HL)
A005  C9                60          RET
Pass 2 errors: 00
Table used:     13 from   107

Hisoft GENA3.1 Assembler. Page      1.
Pass 1 errors: 00
                        10 ;PROGRAMA-2
                        20 ;
A000                    30          ORG  #A000
A000  DD210070          40          LD   IX,#7000
A004  DDCB011E          50          RR   (IX+1)
A008  C9                60          RET
Pass 2 errors: 00
Table used:     13 from   108

Hisoft GENA3.1 Assembler. Page      1.
Pass 1 errors: 00
                        10 ;PROGRAMA-3
                        20 ;
A000                    30          ORG  #A000
A000  3E02              40          LD   A,2
A002  CB27              50          SLA  A
A004  320070            60          LD   (#7000),A
A007  C9                70          RET
Pass 2 errors: 00
Table used:     13 from   108

Hisoft GENA3.1 Assembler. Page      1.
Pass 1 errors: 00
                        10 ;PROGRAMA-4
                        20 ;
A000                    30          ORG  #A000
A000  3E80              40          LD   A,128
A002  CB27              50          SLA  A
A004  320070            60          LD   (#7000),A
A007  C9                70          RET
Pass 2 errors: 00
Table used:     13 from   109

Hisoft GENA3.1 Assembler. Page      1.
Pass 1 errors: 00
                        10 ;PROGRAMA-5
                        20 ;
A000                    30          ORG  #A000
A000  210070            40          LD   HL,#7000
A003  CB2E              50          SRA  (HL)
A005  C9                60          RET
Pass 2 errors: 00
Table used:     13 from   107

Hisoft GENA3.1 Assembler. Page      1.
Pass 1 errors: 00
                        10 ;PROGRAMA-6
                        20 ;
A000                    30          ORG  #A000
A000  3E41              40          LD   A,65
A002  CB2F              50          SRA  A
A004  DC0B00            60          CALL C,IMPAR
A007  D41140            70          CALL NC,PAR
A00A  C9                80          RET
A00B  3E49              90 IMPAR:   LD   A,"I"
A00D  CD5ABB            100         CALL #BB5A
A010  C9                110         RET
A011  3E50              120 PAR:    LD   A,"P"
A013  CD5ABB            130         CALL #BB5A
A016  C9                140         RET
Pass 2 errors: 00
  IMPAR  A00B  PAR   A011
Table used:     35 from   119

Hisoft GENA3.1 Assembler. Page      1.
Pass 1 errors: 00
                        10 ;PROGRAMA-7
                        20 ;
A000                    30          ORG  #A000
A000  210070            40          LD   HL,#7000
A003  3E41              50          LD   A,65
A005  3623              60          LD   (HL),#23
A007  CD5ABB            70          CALL #BB5A
A00A  ED6F              80          RLD
A00C  CD5ABB            90          CALL #BB5A
A00F  ED6F              100         RLD
A011  CD5ABB            110         CALL #BB5A
A014  C9                120         RET
Pass 2 errors: 00
Table used:     13 from   115

Hisoft GENA3.1 Assembler. Page      1.
Pass 1 errors: 00
                        10 ;PROGRAMA-8
                        20 ;
A000                    30          ORG  #A000
A000  210070            40          LD   HL,#7000
A003  3E30              50          LD   A,48
A005  3621              60          LD   (HL),#21
A007  CD5ABB            70          CALL #BB5A
A00A  ED67              80          RRD
A00C  CD5ABB            90          CALL #BB5A
A00F  ED67              100         RRD
A011  CD5ABB            110         CALL #BB5A
A014  C9                120         RET
Pass 2 errors: 00
Table used:     13 from   115
```

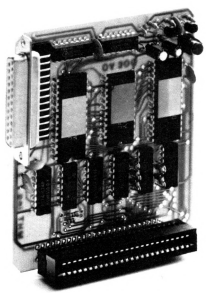
EMPEZAMOS A POTENCIAR TU AMSTRAD

NUEVO INTERFACE RS232

Permite comunicar los ordenadores Amstrad CPC 464, CPC 664 y CPC 6128, con impresoras y plotters con entrada serie, modems y otros ordenadores

CARACTERISTICAS TECNICAS

- Salida serie RS-232C estándar
- Software contenido en Eprom, por lo que no ocupa memoria del usuario. Genera comandos para facilitar el uso desde el basic.



...TAMBIEN PERIFERICOS PARA AMSTRAD

Es un producto desarrollado y fabricado en España por:

MHT INGENIEROS

PRODUCTOS DISTRIBUIDOS: LSB, S.A.

Sánchez Pacheco, 78 - 28002 MADRID - TEL.: 413 92 68

Sin duda alguna

A través de esta sección se pretende resolver, en la medida de lo posible, todas las posibles dudas que **«atormenten»** a todas las personas interesadas en el mundo del AMSTRAD, sean o no poseedores de uno y, si lo son, se encuentren en cualquier nivel de destreza en su manejo.

Semanalmente, aparecen en estas páginas las consultas de la mayor cantidad de usuarios posible; ello redundará en un mejor servicio y en un contacto más estrecho entre todos nosotros a través de la revista.

SIN DUDA ALGUNA está abierta a todos.

Me dirijo a Vds. a través de la sección «SIN DUDA ALGUNA» de la revista **AMSTRAD** Semanal para que me aclaren mis «dudas» entorno a la unidad de disco del **Amstrad**.

Soy el AFORTUNADO poseedor de un CPC 6128 y he leído repetidamente en vuestra revista que está a la venta un interface para unidad de disco de 5.25''. ¿Significa esto que se puede conectar a través de ella cualquier unidad de disco de este formato? ¿Significa también que con ella se podrían correr los programas escritos en este formato en CPM 2.2 o PLUS?

Es mi intención aplicar el CPC 6128 en aplicaciones «serias» y desearía me aclararan este punto ya que con ello podría correr por ejemplo: Dbase II, Friday, etc., ¿es esto así?

A la espera de sus noticias le saluda atte.

Luciano Sancha *(Huelva)*

La interface, efectivamente, existe, pero que nosotros sepamos, todavía no se comercializa en España.

No estamos seguros de si se puede conectar al interface cualquier unidad de disco de 5 1/4 o bien ésta requiere condiciones especiales. Teóricamente, no valdría cualquiera.

*Con esta unidad sí podrían correrse en el **Amstrad** todos los programas CP/M, pero, o bien vienen ya instalados a las particulares características del monitor y teclado **Amstrad**, o bien tendría que instalarlos usted mismo.*

En el mejor de los casos, estos programas llevarían un programa llamado Install o algo así que se lo permitiría, normalmente respondiendo una serie de preguntas sencillas a través de menús. En el peor de los casos, o usted o la casa que le vendiera el programa tendrían que instalarlos ensamblador en ristre.

*Su deseo de emplear el **Amstrad** con aplicaciones serias sólo tiene el límite de la memoria del ordenador que tenga, porque hay programas que se salen fuera de las posibilidades de estas máquinas. De cualquier forma, la instalación de una unidad de disco de 5 1/4 ampliaría considerablemente el monto de programas a los que tendría acceso, siempre bajo CP/M.*

Soy poseedor de un **Amstrad** CPC 464. Mis dos preguntas son:

1. ¿Qué se tiene que hacer para encontrar un programa, en un cassette (donde hay varios programas), sin tener que pasar por todos los primeros?

2. ¿El **Amstrad** tiene un joystick especial, o en cambio son todos estándar?

Gracias por su atención, y perdón por los errores, ya que soy aún un principiante.

Pablo M. Vidallet *(Zaragoza)*

1. La única solución es saber de antemano dónde se encuentra el programa que estás buscando. Bien sabiéndolo a «ojo» y avanzando el cassette hacia allí, o bien usando el cassette con cuentavueltas y apuntando en qué cifra se encuentra cada programa. A todo esto, una de las razones para preferir los discos a los cassettes es precisamente el problema que tú planteas, y que, como ves, tiene difícil solución.

2. Un joystick estándar te servirá perfectamente, ya que no necesitas ningún tipo de interface para que funcione, como es el caso de otros ordenadores. Lo que marca incompatibilidades entre joystick son las interfaces, no el mando de joystick propiamente dicho.

Hola, me he comprado hace poco un **Amstrad** CPC 6128 por lo tanto soy un «novato» en programación, aunque compro la revista semanalmente y presto un especial interés a la sección «Primeros pasos». Me gustaría que me indicárais libros y demás material para sacarle el máximo provecho a mi ordenador.

También me gustaría que me indicárais cómo puedo grabar los programas en disco, ya que en el manual no lo veo claro.

Sin nada más, os agradezco el favor que os demando y deciros que sigáis con la labor ya que me parece estupenda.

Javier Ruiz.

Es difícil recomendar un libro en concreto, porque hay muchos y muy buenos. En tu caso personal, parece que estás empezando, cualquier libro dedicado a principiantes junto con nuestra sección de «Primeros pasos» te servirá perfectamente.

El método para salvar un programa en disco es muy sencillo; teclea, con tu programa en la memoria del ordenador:

SAVE «NOMBRE»

en donde NOMBRE es el que tú hayas elegido para tu programa. Esto te guardará en el disco cualquier programa Basic.

Opcionalmente, puedes escribir:

SAVE «NOMBRE»,A

*para conservar el programa en formato ASCII en lugar de como TOKENS, o sea, números clave que representan secuencias de letras para el **Amstrad**. Por ejemplo, un suponer, la palabra clave DATA se almacena en formato ASCII como D-A-T-A, mientras que como TOKEN en el disco habría un número que representa a esa palabra para el ordenador.*

Los programas en lenguaje máquina se graba de forma muy parecida:

SAVE «NOMBRE», DIRECCION, LONGITUD

en donde NOMBRE ya sabes lo que es, DIRECCION es la dirección de comienzo del programa en máquina, y LONGITUD es la longitud del programa en bytes.

Suponte que tienes un programa en máquina llamado «máquina» cargado en la dirección hexadecimal &A000 y que tiene 100 bytes de longitud; dirías:

SAVE «máquina», &A000, 1000

SORTEO AMSTRAD SEMANAL

¡Nuestro Sorteo ya tiene ganadores! Un CPC 664 y cuatro unidades de disco

El día 17 de enero de 1986 se celebró en las dependencias de la redacción de **AMSTRAD Semanal** el sorteo entre los lectores que en su día rellenaron, y enviaron el cupón-encuesta que apareció en nuestra revista desde el número 1 hasta el número 4.

Los premios consistían en cuatro unidades de disco y un CPC664.

La lista de los afortunados lectores y de los premios que les han correspondido es la siguiente:

PRIMER PREMIO:
AMSTRAD CPC664
Jesús Israel Galvani Díez *(14 años)*
C/ Pintor Lorenzo Casanova, 48-5
30033 Alicante
Tel. 22 83 18

SEGUNDO PREMIO:
UNIDAD DE DISCO
Marta Rosell i Minguella *(15 años)*
C/ Triunf, 6 - esc. 18, 2-1
Barcelona
Tel. 389 40 51

TERCER PREMIO:
UNIDAD DE DISCO
Angel Antonio Gil *(16 años)*
C/ Pepe Magueregui, 1 - 4.º B
26005 Logroño *(La Rioja)*
Tel. 22 76 15

CUARTO PREMIO:
UNIDAD DE DISCO
Manuel Parra López *(17 años)*
C/ Montllor i Pujal, 23
Sabadell *(Barcelona)*
Tel. 725 49 93

QUINTO PREMIO:
UNIDAD DE DISCO
Vicente Asensi Ortega *(13 años)*
C/ Pexis Mencheta, 23 - 2.º
46020 Valencia
Tel. 361 36 03

Mercado común

Con el objeto de fomentar las relaciones entre los usuarios de AMSTRAD, *MERCADO COMUN* te ofrece sus páginas para publicar los pequeños anuncios que relacionados con el ordenador y su mundo se ajusten al formato indicado a continuación.

En *MERCADO COMUN* tienen cabida, anuncios de ventas, compras, clubs de usuarios de AMSTRAD, programadores, y en general cualquier clase de anuncio que pueda servir de utilidad a nuestros lectores.

Envíanos tu anuncio mecanografiado a: **HOBBY PRESS, S.A.**
AMSTRAD SEMANAL.
Apartado de correos 54.062
28080 MADRID
¡ABSTENERSE PIRATAS!

Urgente vendo **Amstrad** CPC 664 con monitor en color 2 discos originales con CP/M, logo, base de datos, etc., y manual en español. Garantía oficial **Amstrad** España a estrenar. Todo por 110.000 pts. Tel. (91) 206 75 21.

Desearía contactar con usuarios del 464/664/6128 para intercambio de programas. Llamar de 10 a 11 de la noche o escribir a: Julián Calero. C/ Cataluña, 16-5.º izqda. Basauri *(Vizcaya)*. Tel. (94) 440 46 88.

Vendo ordenador **Amstrad** 464 con pantalla de fósforo verde. Tengo más de 50 programas y varios libros. Regalo joystick y modulador. Intercambio programas del **Amstrad**. Preguntar por José Luis Soriano. C/ Reus, 6-1. 46009 Valencia. Tel. 349 95 48.

Por necesidades económicas vendo ordenador **Amstrad** CPC-664 con unidad disco 3'' incorp., monitor Color 14'' y progr. CP/M, Logo, Base Datos, Proces. Textos, Tasprint, Tascopy y 6 progrs. más, todos en discos. Comprado el 5-10-85 con garantía oficial. Precio mínimo 98.000 pts. al contado (aceptaré la mejor oferta). Urge venta. A. Moreno. C/ Riera Alta, 43, at. 3.ª 08001 Barcelona.

Desearía contactar con usuarios de **Amstrad** de Zaragoza de forma especial para comprar, vender, intercambiar, juegos, información ideas sobre el ordenador. (Programas prácticos, juegos...). Compraría joystick en buen estado y barato. Mi dirección: Pilar Gayán Claro. C/ San Ignacio de Loyola, 1 - 6.º A. Tel. (976) 22 23 61. Llamar a ser posible 14 a 17 y 20 a 22,30 H.

Desearía contactar con propietarios de CPC 664 y CPC 6128 para formar un club de usuarios en Zaragoza y provincias limítrofes. Mariano de la Iglesia. Tel. (976) 56 01 07. Zaragoza.

Vendo libro «Hacia la inteligencia artificial con **Amstrad**, por tener el listado mal, por 500 pts. Escribir a: José Martínez Sánchez. Avda. Andalucía, 9. Tel. (956) 25 37 86.

Desearía contactar con usuarios CPC 464 para intercambio de programas, trucos, ideas, información, etc. Francisco José Sampedro Luján. Avda. Conde Lumiares, 33 - 3.º C. 03010 Alicante. Tel. (965) 25 11 76. LLamar de 2 a 3 excepto fines de semana.

Vendo impresora Compute Mate-100 modelo CPA-80. A estrenar. Doble tracción-fricción, 100 CPS cable centronic. Funda. Garantía. Muy barato. Tardesnoches. Ana Saiz. Madrid. Tel. (91) 200 57 01.

Vendo Amsword I «trat. de textos» (1.700 pts.), Amsbase «base de datos» (1.700 pts.), Amscalc «hoja de cálculo» (3.500 pts.), Harrier-Attack (800 pts.) y Master Chess «ajedrez» (1.500 pts.), todos originales y en sus fundas con los correspondientes manuales. Lote completo por 8.000 pts. Tel. (942) 33 13 52.

MICRO-1

Duque de Sexto, 50. 28012 Madrid
Tels. (91) 274 53 80-276 96 16

SOFTWARE: ¡¡GRATIS 1 BOLIGRAFO DE ACERO CON RELOJ INCORPORADO!!

	Ptas.		Ptas.
Bounty Bob	2.300	Bruce Lee	2.300
Fighting Warrior	2.100	Yier Kung Fu	2.300
Foupack	3.890	Exploding Fist	2.300
Combat Lynx	2.100	Southern Belle	2.300
Dummy Run	2.100	Dragontorc	2.300
Raid	2.300	Rocky	2.100
Match Day	2.300	World Series Baseball	2.100
Map Game	2.750	Dambusters	2.300
Hypersports	2.300	Ajedrez Tridimensional	1.975

IMPRESORAS: ¡¡20% DE DESCUENTO SOBRE P.V.P.!!

Lápiz óptico DK'Tronics	4.850	Diskette 3"	1.050
Tapa metacrilato AMSTRAD	1.975	Cinta C-15 (especial)	85
Toshiba MSX 64 K	39.900	Cassette Especial	5.295

Joystick Quick Shot II
2.495 ptas.

Joystick Quick Shot I
1.995 ptas.

Joystick Quick Shot V
2.995 ptas.

Increíbles precios para tu AMSTRAD 464 y 6128 (verde y color). (Llámanos y te asombrarás)

PC Compatible IBM 256 K
Monitor Fósforo Verde
2 Bocas Diskette 360 K
279.000 ptas.

Sabrewulf + Decathlon + Beach Head + Jet Set Willy
2.500 ptas.

Commodore 64: 42.900 ptas.
Commodore 128: 74.900 ptas.

El pedido te lo enviamos URGENTEMENTE contra-reembolso SIN NINGUN GASTO DE ENVIO, LLAMANDO a los teléfonos: (91) 276 96 16-274 53 80 o escribiendo a MICRO-1. Duque de Sexto, 50. 28012 Madrid.

O SON SEIKOSHA ...

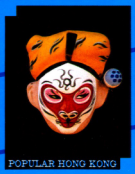

CARNAVAL SITGES


OPERA CHINA


DIOS NEPALI

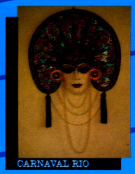

SEIKOSHA GP


OPERA CHINA


POPULAR HONG KONG


SEIKOSHA SP


POPULAR CHINA

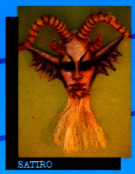

CARNAVAL RIO


SEIKOSHA MP


CERAMICA MANISES


OPERA JAPONESA


SEIKOSHA BP


SATIRO


POPULAR JAPON

...O SON MASCARAS

GP-50 •	La pequeña 40 cps. Papel normal con interface paralelo, serial y Spectrum	17.990 ptas.
GP-700 *	La de color 50 cps. 7 colores. 80 columnas. Tracción y fricción. Papel de 10 pulgadas	64.990 ptas.
SP-1.000 *	La programable 100 cps.24 cps en alta calidad 96 cart. programables en RAM. Introductor hoja a hoja.♦	64.990 ptas.
SP-1.000AS	La programable 100 cps.24 cps en alta calidad con interface RS-232. Introductor hoja a hoja.♦	59.900 ptas.
MP-1.300AI	La polivalente 300 cps, 60 cps en alta calidad, interface paralelo y RS-232. Introductor hoja a hoja.♦&	119.900 ptas.
BP-5.200 *	La de oficina 200 cps, 106 en alta calidad.Buffer 4K.Carro de 15".Tracción y fricción.♦	199.900 ptas.
BP-5.420 *	La más rápida 420 cps. 106 cps en alta calidad. Buffer de 18K. Paralelo y RS-232.♦	299.900 ptas.

Interfaces: Serie RS-232C, Spectrum, IBM, COMMODORE, MSX, QL, Apple Macintosh, HP-IB
♦ Disponen de introductor automático de documentos opcional.
& Dispone de Kit opcional de color
Nota: I.V.A. 12%, no incluido en los precios arriba indicados

* con interface paralelo
• con interface Spectrum

Avda. Blasco Ibáñez, 116
Tel. (96) 372. 88. 89
Telex 62220 - 46022 VALENCIA

Muntaner, 60-2.º-4.ª
Tel. (93) 323. 32. 19
08011 BARCELONA

Agustin de Foxá, 25-3.º-A
Tels. (91) 733. 57. 00 - 733. 56. 50
28036 MADRID

DiRAC