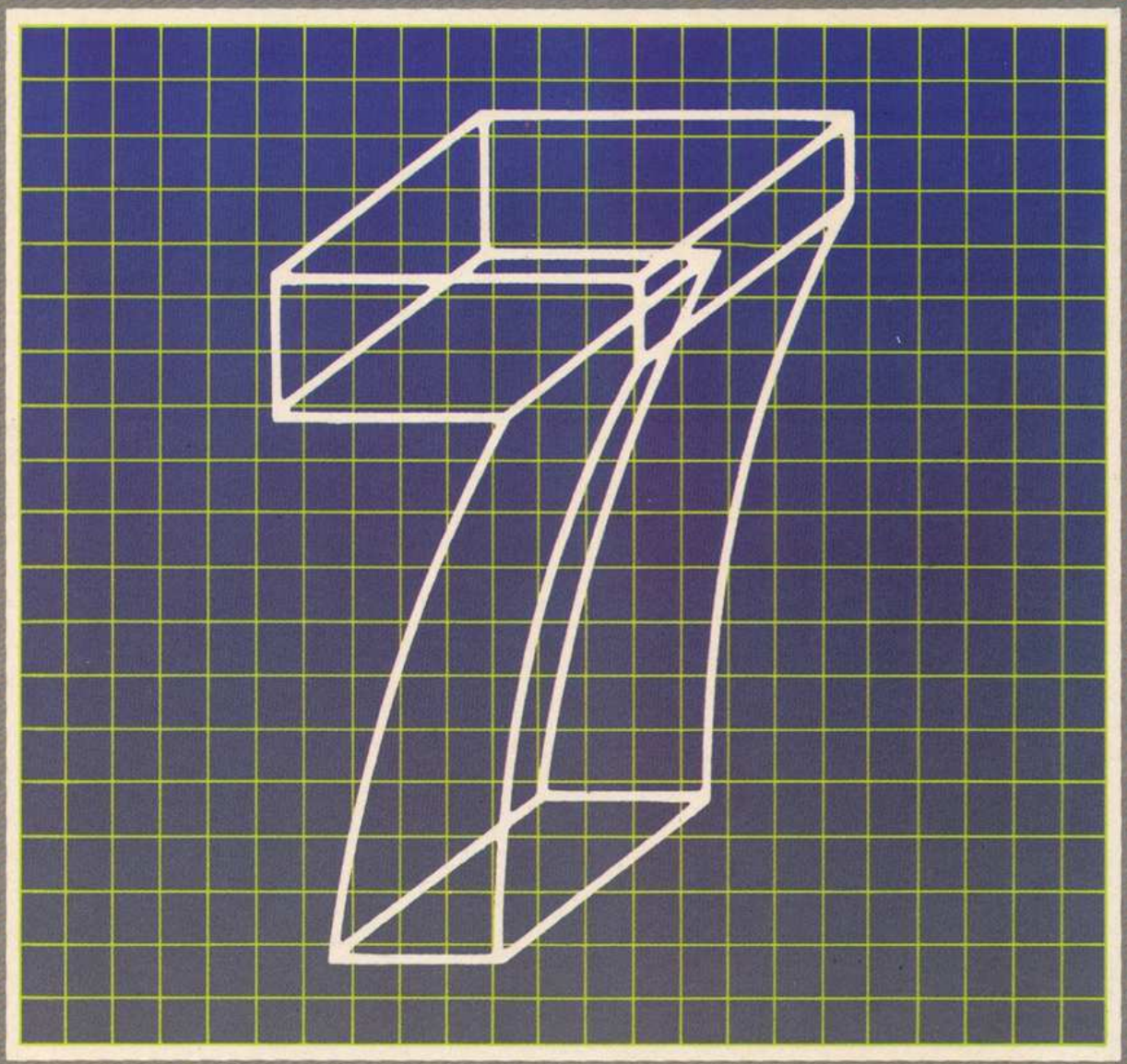
BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD, IBM, SPECTRUM, COMMODORE Y MSX



PROGRAMAS EDUCATIVOS, DE GRAFICOS Y DE UTILIDAD

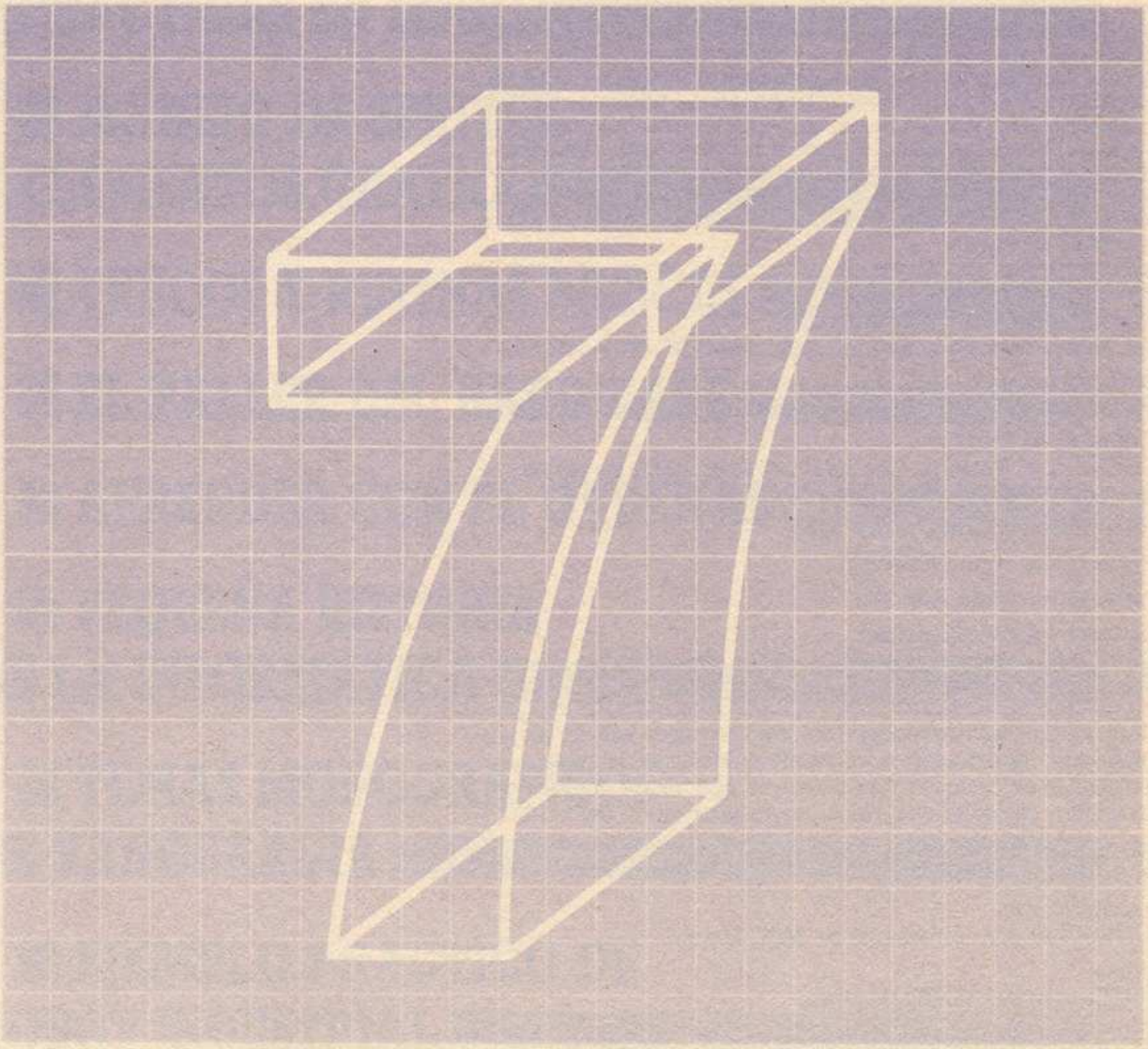
EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

TALLER DE INFORMATICA

PROGRAMAS VALIDOS PARA AMSTRAD, IBM, SPECTRUM, COMMODORE Y MSX



EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:
RICARDO ESPAÑOL CRESPO.

Gerente:
ANTONIO G. CUERPO.

Directora de producción:
MARIA LUISA SUAREZ PEREZ.

Director de la colección:
JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:
BRAVO-LOFISH.

Maquetación:
D. S. M.

Dibujos:
JOSE OCHOA

Tomo 7. «Experiencias prácticas en logo», Carlos Albert, Técnico de Informática; Adoración Llena Jubero, Técnico de Informática. «Manejo de sprites y elementos gráficos», «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprende con el ordenador», AULA DE INFORMATICA APLICADA: Soledad Tamariz-Martel, Diplomada en Telecomuniación; Francisco Blanca, Diplomado en Telecomunicación; Alejandro Marcos, Licenciado en Química; Fernando Suero, Diplomado en Telecomunicación. «El Taller del Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», «Vocabulario de Informática», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:
Sor Angela de la Cruz, 24-7.º G. Teléf. 279 40 36. 28020 Madrid.

Publicidad:
Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:
COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.
Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:
CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.
Buenos Aires - 1.290. Argentina.

ISBN del tomo: 84-7688-064-2
ISBN de la obra: 84-7688-047-2

Fotocomposición:
ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:
MATEU CROMO. PINTO (Madrid).

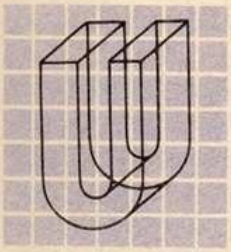
Depósito legal: M-43.185-1986
Printed in Spain - Impreso en España.

Suscripciones y números atrasados:
Ediciones Siglo Cultural, S. A.
Sor Angela de la Cruz, 24-7.º G. Teléf. 279 40 36. 28020 Madrid.

Marzo, 1987

INDICE

EXPERIENCIA Y PRACTICAS EN LOGO



N procedimiento lo definimos de la siguiente manera:

```
? PARA nombre
>
>
>
. (órdenes)
.
> FIN
```

Si defines un procedimiento que dibuje un rectángulo:

```
? PARA RECTAN
> REPITE 2 [AV 20 GD 45 AV 40 GD 90]
> FIN
```

Y ahora lo ejecutas:

```
? RECTAN
```

Aparecerá lo siguiente:

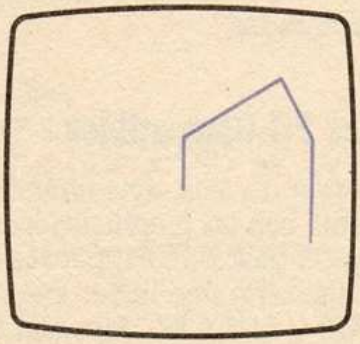


Fig. 1

Observa que el dibujo obtenido no es un rectángulo.

Si repasas el procedimiento, verás que nos hemos equivocado y en vez de girar a la derecha 90 grados la primera vez, hemos girado 45.

Vamos a ver cómo podemos solucionar este pequeño problema.

Una de las formas sería repitiendo todo otra vez. Si intentas definir otra vez RECTAN, no podrás hacerlo, ya que estás utilizando el mismo nombre. Lo único que podrías hacer sería definir el mismo procedimiento con otro nombre.

Otra forma sería obteniendo el programa de las órdenes que forman el procedimiento. Con el cursor nos desplazaríamos hasta la posición donde se ha cometido el error y lo corregiríamos.

Esto lo podemos hacer, ya que el Logo nos permite obtener este programa y corregirlo, añadir quitar o cambiar todo lo que queramos. Lo primero que tendremos que hacer es dar la orden necesaria para obtener el programa.

Esta es:

```
EDITA "nombre de procedimiento
```

Y de forma abreviada:

```
ED "nombre de procedimiento
```

Con esta orden entramos en el editor del LOGO. Para salir de él, basta con pulsar la tecla F1, en el caso de que estés trabajando con un ordenador MSX o un PC-compatible. Si tu ordenador es un SPECTRUM, pulsa las teclas E MODE y C.

EDITOR LOGO

Al dar la orden anterior lo primero que sucede es que desaparece la pantalla que teníamos en ese momento y aparece otra, en donde viene la frase, EDITOR LOGO, y el programa de todas las órdenes que pertenecen al

Da a los procedimientos un nombre que identifique lo que hacen.

procedimiento que hemos pedido que nos edite, encabezadas con PARA y el nombre del procedimiento. Ahora ya no aparece ni el signo de interrogación ni el signo >.

Siempre que queramos recuperar un procedimiento, para efectuar algún cambio o simplemente para verlo, tendremos que editarlo.

Una vez que lo tenemos editado, podemos desplazarnos con el cursor a cualquier posición y realizar el cambio necesario.

En el editor solamente aparecen las órdenes y no podremos ver lo que hace ninguna de ellas. Esta pantalla está reservada para teclear nuestras órdenes.

Cuando hayamos realizado las modificaciones necesarias y queramos ver lo que hacen, tendremos que salir del editor. Esto lo conseguimos pulsando la tecla F1. Nada más pulsarla, pasamos al modo de pantalla en el que estábamos anteriormente, sin que nada de lo que en ella se encontraba se haya borrado.

Si queremos corregir el error que hemos cometido en el procedimiento que hemos definido antes, lo haríamos de la siguiente forma:

Primero lo editamos:

```
? ED "RECTAN
```

Obtendríamos:

```
PARA RECTAN
REPITE 2 [AV 20 GD 45 AV 40 GD 90]
FIN
```

Con las teclas de desplazamiento, colocaríamos el cursor en la posición en donde está el número 45. Lo borraríamos y en su lugar teclearíamos 90. A continuación, saldríamos del editor. Una vez fuera, volveríamos a ejecutarlo:

```
? RECTAN
```

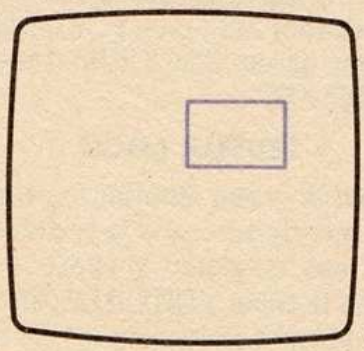


Fig. 2

Ahora, nuestro rectángulo aparece bien dibujado.

Si lo que quieres es borrar algo, hazlo con la tecla DELETE o DEL, o bien con la tecla BS, que también podrás encontrar representada por ← en algunos ordenadores. Pero recuerda que la tecla DELETE o DEL borra el carácter que se encuentra a la derecha a partir de la posición en la que esté el cursor, y que la tecla BS borra el carácter que está a la izquierda.

Si lo que quieres es añadir, no hace falta que pulses la tecla INS (insertar), ya que el editor del Logo va desplazando hacia la derecha todos los caracteres a medida que vamos tecleando.

Ya hemos dicho anteriormente que para salir del editor Logo hay que pulsar una o unas teclas, que dependiendo del ordenador y de la versión de Logo con la que estemos trabajando, éstas son unas determinadas. Podemos salir del editor Logo, bien tomando en cuenta las modificaciones que se hayan realizado o bien sin tomarlas en cuenta.

Vamos a ver cómo se realizan estos procesos dependiendo del ordenador.

Para los MSX

Si pulsamos la tecla F1, saldremos del editor y todas las modificaciones realizadas serán tomadas en cuenta. Por el contrario, si pulsamos F5, saldremos, pero las modificaciones serán ignoradas.

Para los PC-Compatibles

La forma de salir actualizando nuestro procedimiento con las modificaciones realizadas es también pulsando F1, pero la forma de salir sin actualizarlo puede ser mediante F10, CTRL y STOP o ALT y F2, dependiendo de la versión Logo.

Para el Spectrum

Si pulsamos las teclas CAPS y BREAK/SPACE, salimos del editor sin tomar en cuenta las modificaciones. Para que las tome en cuenta hay que pulsar las teclas E MODE y C.

El Logo posee su propio editor.

Existen en todos los casos teclas que realizan funciones específicas dentro del editor, como borrar líneas completas, posicionarnos en un lugar determinado con el cursor, etcétera. En cada caso son diferentes y bastante numerosas, por lo que ya las iremos viendo poco a poco.

A continuación, realizaremos unos dibujos utilizando los procedimientos. Observa que realizamos los dibujos con más de un procedimiento.

```
? PARA MOLINO
> BP PM
> OT BL
> PONCL 2
> REPITE 2 [AV 65 GD 90 AV 42 GD 90]
> AV 65 GD 45
> AV 30 GD 90 AV 30
> SL
> PONPOS [18 0]
> PONRUMBO 0
> BL
> REPITE 2 [AV 25 GD 90 AV 10 GD 90]
> SL PONPOS [20 65]
> BL
> PONRUMBO 45
> ASPA
> PONRUMBO 135
> ASPA
> PONRUMBO 225
> ASPA
> PONRUMBO 320
> ASPA
> FIN

? PARA ASPA
> AV 2 GI 90 AV 2 GD 90
> REPITE 2 [AV 30 GD 90 AV 4 GD 90]
> FIN
```

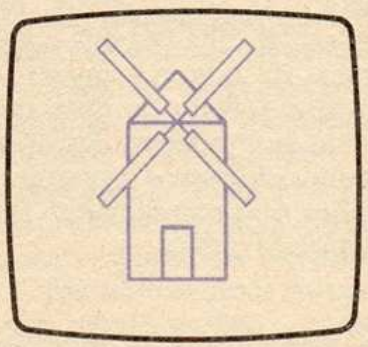


Fig. 3

Para que se realice este dibujo llama desde fuera del editor al procedimiento principal.

```
? MOLINO
```

Recuerda que si posees un ordenador SPECTRUM, no puedes dar la orden PM (pantallamixta); por tanto, no la teclees. Debes cambiar la orden OT (ocultatortuga) por ET (escondetortuga) y la orden BL (BAJALAPIZ) por CL (CONLAPIZ).

Si antes de empezar a teclear las órdenes que definen un dibujo analizas de qué se compone, en este caso el molino, observas que hay algo que se repite, en concreto las aspas, que son idénticas, que lo único que varía de una a otra es la colocación; por tanto, eliminas trabajo definiendo un procedimiento que dibuje un aspa.

Desde el cuerpo principal, por ejemplo, un procedimiento, posicionas a la Tortuga en el lugar donde debe ir cada aspa y llamas al procedimiento que la dibuja.

Aparte de ahorrarte trabajo, consigues estructurar el programa de forma clara y sencilla.

En este otro ejemplo también hacemos llamadas a procedimientos dentro de otros procedimientos:

```
? PARA CASTILLO
> BP OT PM
> SL PONPOS [-35 0]
> BL
> AV 70 REPITE 3 [DIENTE]
> GD 90 AV 5 GD 90 AV 70 RE 20
> SL PONPOS [0 50]
> BL
> REPITE 7 [DIENTE]
> GD 90 AVV 5 GI 90 RE 50 AV 70
> REPITE 3 [DIENTE]
> GD 90 AV 5
> GD 90 AV 70
> GD 90 AV 145
> RE 60 GD 90 AV 15
> REPITE 18 [AV 2 GD 10] AV 17
> SL PONPOS [-15 55]
> VENTANAS
> SL PONX 100
> VENTANAS
> FIN
```

En el editor Logo no se ejecuta ninguna orden, sólo se escribe o corrige.

```
? PARA DIENTE
> PONRUMBO 90
> AV 5 GD 90 AV 5 GI 90 AV 5 GI 90 AV 5
> FIN

? PARA VENTANAS
> BL
> REPITE 4 [AV 10 GD 90]
> FIN
```

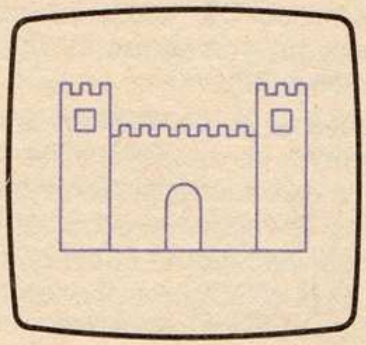


Fig. 4

Para ejecutar este conjunto de órdenes teclea:

```
? CASTILLO
```

Veamos algo más sobre la definición de los procedimientos.

El mensaje que recibimos una vez que hemos terminado de definir un procedimiento varía según el ordenador con el que estemos trabajando.

En los MSX es: nombre procedimiento DEFINIDO.

En los PC-COMPATIBLES: QUEDA DEFINIDO nombre de procedimiento.

Y en otras versiones: ACABAS DE DEFINIR nombre de procedimiento.

En el SPECTRUM: nombre de procedimiento DEFINIDO.

El nombre de procedimiento es el que se ha especificado en PARA.

En todos ellos el significado es el mismo; se ha definido un procedimiento determinado con una serie de órdenes bajo el nombre que aparece.

Respecto al nombre que se da al procedimiento, ya hablamos de la limitación que existe en la longitud. En cuanto a qué nombre puede utilizarse vimos que en principio podía ser cualquiera, pero existen también limitaciones.

Date cuenta que cuando escribimos el nombre de un procedimiento para que se realice, la forma de hacerlo es la misma que cuando ponemos cualquier otra orden del Logo. Por tanto, es fácil suponer que nuestro procedimiento está actuando como otra orden que realiza a su vez una serie de órdenes.

Si definimos un procedimiento con el nombre de alguna de las primitivas de Logo, nos aparecerá un mensaje de error, ya que no podemos utilizar ninguno de estos nombres.

Sería imposible, por ejemplo, hacer lo siguiente:

```
? PARA AV
```

ya que AV es una primitiva y tiene una función determinada que realizar.

Sería perfectamente válido:

```
? PARA AV2
> AV 20
> FIN
```

Cada vez que introduzcamos AV2, la Tortuga avanzará 10 pasos.

Existe otra forma de definir nuestros procedimientos. Por ejemplo, si queremos definir un procedimiento que dibuje una casa, podemos realizarlo de la siguiente forma:

En vez de dar la orden PARA CASA, nos vamos directamente al editor. Si introducimos:

```
? ED "casa
```

Aunque CASA todavía no está definido, nos aparece lo siguiente en el editor:

```
PARA CASA
FIN
```

Podemos empezar a dar las órdenes, pero en este caso directamente las estamos dando en el editor. Una vez que hayamos acabado, salimos del editor y nuestro procedimiento quedará definido, al igual que ocurría haciéndolo de la otra forma. Siempre que salimos del editor, volvemos a recibir el mensaje de, "nombre de procedimiento DEFINIDO".

Si damos la orden:

```
? ED
```

Pasaremos al editor y en él aparecerá el último procedimiento que hayamos editado.

No puede usarse el nombre de una primitiva para definir un procedimiento.

Si no hubiese ningún procedimiento, el editor aparecería en blanco.

Si damos la orden:

? ED"

Pasaríamos al editor y en él aparecería:

PARA
FIN

Cuando saliésemos de él, aparecería un mensaje diciéndonos que faltan datos para PARA.

Resumiendo, en el editor podemos escribir todo lo que queramos, pero para que lo escrito constituya un procedimiento, a la fuerza tiene que aparecer PARA y FIN.

Guardar y cargar procedimientos II

Guardar procedimientos

Si vamos a almacenar información (en nuestro caso procedimientos) en un soporte externo, sea cual sea, deberá estar debidamente preparado para poder aceptarla.

Por lo general, podemos guardar los procedimientos en dos tipos de soportes, cintas de cassette o discos. Dependiendo del ordenador, éste tendrá un sistema u otro, y en algunos casos ambos.

Si utilizamos un disco para guardar los procedimientos, éste deberá estar formateado. Para hacer esto, mira el Manual de tu ordenador, donde se explica la forma correcta de hacerlo. Este es un proceso general y no particular del Logo.

Un disco siempre tiene que estar formateado para poder almacenar en él cualquier tipo de información.

Si, por el contrario, utilizas la cinta de cassette, ésta no necesita ninguna preparación previa. Simplemente debes cerciorarte de que no vas a guardar nada encima de algo que te sea útil. Pon siempre la cinta al principio de la zona de grabación y sigue las instrucciones que te dé el ordenador.

Si tu ordenador tiene más de una unidad de disco, o bien puede utilizar el disco y la cinta u otro tipo de soporte; tendremos que determinar, en primer lugar, el dispositivo en donde queremos que se guarde nuestra información. Esto es posible, ya que el Logo posee una orden con la cual activamos el soporte con el que queramos trabajar. Se trata de la orden:

PONDISCO disco

Donde disco puede ser un número o una letra, dependiendo de cómo llame cada ordenador a las unidades con las que trabaja.

Vamos a ver de qué dispositivos disponen algunos ordenadores y cómo podemos elegir uno u otro.

Ordenadores MSX

Dispone de disco y de cinta de cassette.

Si queremos usar el disco y tenemos más de una unidad, tendremos que determinar en cuál queremos que nos guarde. Lo determinaremos con:

PONDISCO n

n puede ser como mínimo 1 y como máximo 9, y activará la unidad que determina.

Si queremos usar la cinta tendremos que dar la orden:

PONDISCO "C

"C es el código reservado para activar el cassette y desactivar las unidades de disco.

Ordenadores PC-Compatibles

Disponen exclusivamente de disco, pudiendo ser éste rígido o flexible. En este caso se activará una unidad determinada de la siguiente forma:

PONDISCO A, B, C,...

A, B, C,... corresponden a las diferentes unidades que posea el ordenador.

Lo normal es tener dos unidades, bien de dos discos flexibles o bien de un disco flexible y uno rígido. La C suele estar reservada para el disco rígido.

La forma correcta de activar un determinado disco es:

PONDISCO "A:

Spectrum

Dispone de cinta de cassette y de microdrive.

Si queremos utilizar el cassette, lo activaremos con:

PONDISCO 0

y si queremos utilizar el microdrive:

PONDISCO n

Puedo definir los procedimientos directamente en el editor Logo.

Activamos el microdrive que determina n, pudiendo ser del 1 al 8.

El microdrive tiene que estar, al igual que los discos, formateado.

Cargar procedimientos

A la hora de cargar en la memoria del ordenador algún fichero de procedimientos determinados, ya vimos cómo se hacía.

En este caso, también puede determinarse la unidad desde la que queremos cargar. Para ello utilizaremos la misma orden anterior PONDISCO, la cual nos activará la unidad deseada.

Para que practiques en el manejo de la tortuga y al mismo tiempo entiendas mejor la utilidad de los procedimientos, a continuación hemos preparado una serie de procedimientos que ejecutados o llamados en las posiciones de pantalla adecuadas componen el dibujo de una fotografía urbana:

```
? PARA DIBUJO
> BP
> PM
> OT
> PONFONDO 1
> MARCO
> PONCL 2
> SL PONPOS [-125 -52]
> CALLE
> PONRUMBO 0
> SL PONPOS [-30 -65]
> AUTOBUS
> SL PONPOS [-130 -50]
> VALLA
> PONRUMBO 0
> SL PONPOS [95 -50]
> BUZON
> PONRUMBO 0
> SL PONPOS [-75 30]
> CASA
> PONRUMBO 270
> SL PONPOS [70 -18]
> ESPERA 5000
> FIN
```

Este primer procedimiento va llamando a todos los restantes, marcando en qué posición de la pantalla deben plasmarse los dibujos que estos otros procedimientos realizan.

Introdúcelos todos estos módulos que componen el dibujo:

```
? PARA MARCO
> SL
> PONPOS [-125 -88]
> BL
> REPITE 2 [AV 176 GD 90 AV 256 GD 90]
> FIN
```

El dibujo lo hemos enmarcado dentro de este cuadro, para así delimitar las posiciones de la pantalla. Con esto conseguimos que el dibujo pueda realizarse en cualquier tipo de pantalla con resolución mayor o igual a 256 × 176 puntos:

```
? PARA CALLE
> GD 90
> REPITE 2 [BL AV 95 SL AV 114 BL AV 47 SL
  RE 256 GD 90 AV 10 GI 90]
> FIN
```

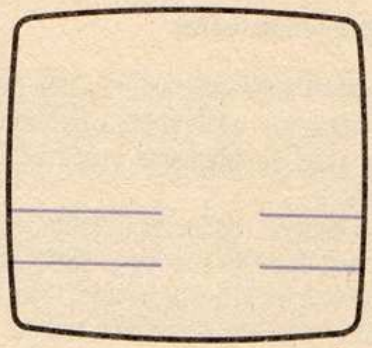


Fig. 5

```
? PARA AUTOBUS
> BL
> AV 60 GD 40 AV 10 GD 50
> AV 100 GD 40 AV 10 GD 50
> RUEDA
> AV 40
> RUEDA
> AV 14
> SL
> GD 90 AV 40 GI 90 AV 20 GD 90
> REPITE 3 [SL GD 90 AV 20 GI 90 BL REPITE
  4 [AV 20 GD 90]]
> GD 90 AV 20 GI 90 AV 20 GD 90
> REPITE 2 [AV 20 GD 90 AV 60 GD 90]
> SL AV 30
> BL REPITE 4 [AV 20 GD 90]
> FIN
```

Las ruedas del autobús las dibujamos con otro procedimiento:

Un procedimiento puede contener a otro.

```
? PARA RUEDA
> GI 90
> REPITE 36 [AV 2 GD 10]
> GD 90 SL AV 25
> BL
> FIN
```

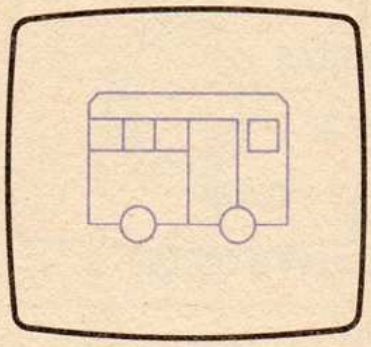


Fig. 6

```
? PARA VALLA
> REPITE 5 [SL PONRUMBO 90 AV 16 GI 90 BL
  REPITE 2 [AV 40 GD 90 AV 8 GD 90]]
> SL
> GD 90 AV 12
> GI 90 AV 8
> GI 90
> BL
> REPITE 2 [AV 80 GD 90 AV 6 GD 90]
> SL
> GD 90 AV 16 GI 90
> BL
> REPITE 2 [AV 80 GD 90 AV 6 GD 90]
> FIN
```

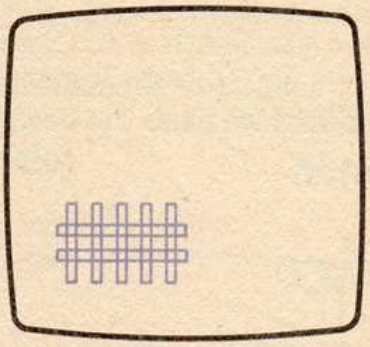


Fig. 7

```
? PARA CASA
> BL
> AV 40 GD 90
> AV 37
> RE 37
> GI 40 AV 25
> GD 80 AV 25
> GD 50 AV 40
> GD 90 AV 15
> GD 90 AV 10
> REPITE 18 [AV 1 GI 10]
> AV 11 GD 90
> AV 30 GD 90
> AV 28 GD 90
> AV 17
> SL AV 5
> BL
> REPITE 4 [AV 7 GI 90]
> SL AV 21
> BL
> REPITE 4 [AV 7 GI 90]
> FIN
```

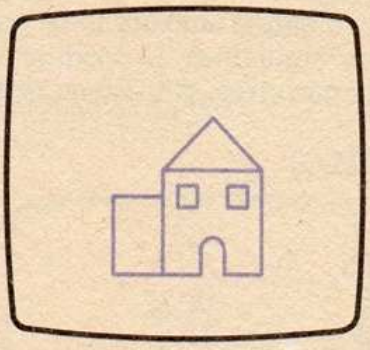


Fig. 8

```
? PARA BUZON
> BL
> AV 5 GD 30
> AV 4 GD 60
> AV 27
> RE 27
> GI 90 AV 30
> GI 90 AV 3
> GD 140 AV 22
> GD 80 AV 22
> GD 140 AV 32
> RE 29
> GI 90 AV 30
> GI 30 AV 4
> GD 30 AV 5
> GD 30 AV 31
> SL
> GD 90 AV 27
> GD 90 AV 10
> BL
> REPITE 2 [AV 10 GI 90 AV 4 GI 90]
> FIN
```

Puedo salir del editor tomando en cuenta las modificaciones o sin tomarlas.

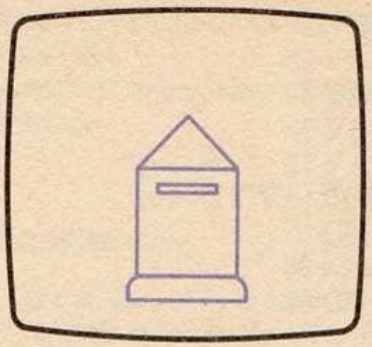


Fig. 9

Una vez tengas todos los procedimientos en memoria, teclea el nombre del primero, y el dibujo irá pintándose en la pantalla:

```
? DIBUJO
```

Si componer un dibujo como lo hemos hecho en el ejemplo anterior te ha gustado, introduce los siguientes procedimientos que, unidos unos con otros, dibujan un reloj despertador:

```
> PARA RELOJ
> BP PM
> OT
> PONCL 3
> ESFERA
> SAETAS
> RING
> PATAS
> FIN
```

Igual que hacíamos en el dibujo del paisaje anterior, uno de los procedimientos lo definimos para que una a los restantes, en este caso no marcamos las posiciones en donde queremos que se dibujen las partes del reloj, porque éstas se definen dentro de cada uno de los procedimientos:

```
? PARA ESFERA
> SL
> PONPOS [4 -40]
> GI 90
> BL
> REPITE 36 [AV 7 GD 10]
> GD 90
> SL
> RE 6 GI 90
> BL
> REPITE 36 [AV 8 GD 10]
> SL
> CENTRO
> REPITE 12 [AV 30 BL AV 4 SL RE 34 GD 30]
> FIN
```

ESFERA se encarga de dibujar la esfera del reloj y las líneas que representan las horas.

Ahora definimos el dibujo de las agujas del reloj:

```
? PARA SAETAS
> BL
> RE 2 AV 21
> SL
> GD 90 AV 3 GI 180
> BL
> REPITE 3 [AV 6 GD 120]
> SL
> CENTRO
> GD 270
> BL RE 2 AV 16
> SL GD 90 AV 3 GI 180
> BL
> REPITE 3 [AV 6 GD 120]
> FIN
```

RING contiene a las órdenes con las que pintaremos el timbre:

```
? PARA RING
> SL
> CENTRO
> PONRUMBO 0
> AV 45
> BL AV 5 GD 90 AV 17
> RE 34 GI 90
> REPITE 19 [AV 3 GD 10]
> FIN
```

Y, por último, este procedimiento con el que dibujaremos las patas del despertador:

```
? PARA PATAS
> SL
> CENTRO
> GD 160 AV 47
> BL
> AV 10
> SL
> CENTRO
> GD 200 AV 47
> BL
> AV 10
> GI 110 AV 38
> FIN
```

Puedo entrar en el editor sin pedir la edición de un procedimiento.

Para ejecutarlo, teclea el nombre del procedimiento que engloba a todos los demás:

? RELOJ

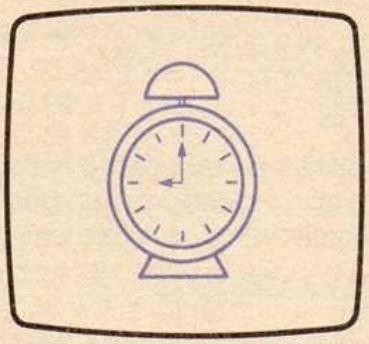


Fig. 10

Cuadro resumen

– EDITA "nombre de procedimiento
ED "nombre de procedimiento

Nos introduce en el editor Logo, mostrándonos todas las órdenes que componen el procedimiento que se determina en nombre de procedimiento.

Si no se especifica el nombre del procedimiento, aparecerá el último que se haya editado. En el caso que no se haya definido ninguno, pasaremos al editor sin obtener ninguna relación de órdenes.

– PONDISCO n
PONDISCO "x
PONDISCO "x:

Activa la unidad que determina n, si ésta recibe un número para ser identificada, o bien la que determina x, si recibe una letra.

Ejercicios

1. *Haz el siguiente dibujo:*

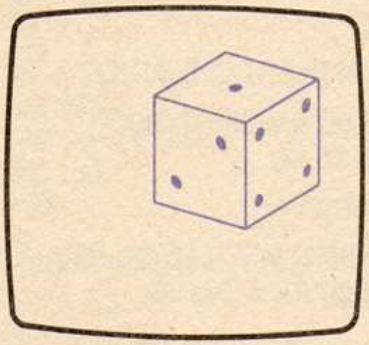


Fig. 11

2. *¿Son correctos los siguientes procedimientos?*

```
? PARA A
> AV 10 GD 90
> REPITE 4 [AV 10 GD 90]
> FIN

? PARA B
> REPITE 4 [A]
> FIN

? PARA C
> REPITE 4 [B]
> FIN
```

3. *Haz el siguiente dibujo:*

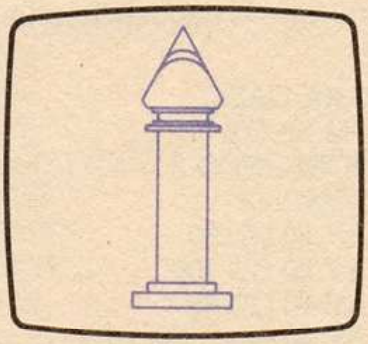


Fig. 12

4. *Haz un procedimiento que haga el siguiente dibujo. Este procedimiento deberá contener a su vez cuatro diferentes e independientes y cada uno debe realizar lo siguiente:*

1.º Que haga el recuadro.

2.º Que haga los ejes.

3.º Que haga la escala horizontal.

4.º Que haga la escala vertical.

Las dimensiones que te damos están expresadas en puntos.

Puedo determinar el soporte en el que quiero guardar los procedimientos.

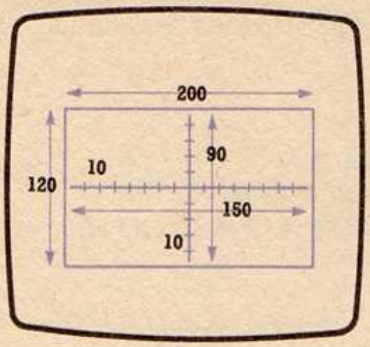


Fig. 13

5. *Son correctas las siguientes órdenes:*
   - PARA PARA
   - EDITA "FF
   - ED CASA
   - GUARDA "LAV
   - GUARDA "REPITE
   - PARA LIBRO

Solución a los ejercicios

1:

Dibujamos el dado utilizando dos procedimientos:

```
? PARA DADO
> PM BP OT
> PONFONDO 2
> PONCL 2
> AV 50 GD 80
> AV 50 GD 100
> AV 50 GD 80
> AV 50
> PONRUMBO 290
> AV 50
> PONRUMBO 0
> AV 50
> PONRUMBO 110
> AV 50
> RE 50
> GI 30 AV 50
> GD 30 AV 50
> PONRUMBO 0
> SL PONPOS [-6 63]
> CIRCULO
> SL PONPOS [-16 43]
> CIRCULO
> SL PONPOS [-43 23]
> CIRCULO
> SL PONPOS [6 42]
> CIRCULO
> SL PONPOS [6 10]
> CIRCULO
> SL PONPOS [34 15]
> CIRCULO
> SL PONPOS [34 47]
> CIRCULO
> FIN
```

En este procedimiento hacemos varias llamadas a otro procedimiento, CIRCULO, que, como su nombre indica, pinta círculos.

```
? PARA CIRCULO
> BL
> REPITE 36 [AV 1 GD 10]
> FIN
```

El procedimiento principal en este caso es DADO.

```
? DADO
```

2:

Los tres procedimientos son correctos.

El A, dibuja un cuadrado después de avanzar 10 y girar a la derecha 90 grados.

El B, repite 4 veces el procedimiento A.

El C, repite 4 veces el procedimiento B.

Tanto el B como el C hacen una llamada a otro procedimiento, lo cual es perfectamente válido en el Logo, incluso cuando el procedimiento al que se llama tiene una llamada a otro distinto, como en el caso del C.

3:

```
? PARA ALFIL
  INICIALIZACION
> PM
> SL
> BP
> OT
  CENTRANDO DIBUJO
> GI 90 AV 35 GD 90
> RE 60 BL
  DIBUJO
> REPITE 2 [AV 10 GD 90 AV 70 GD 90]
> GI 90 RE 5 GD 90
```

Si no se especifica el nombre de un procedimiento a la hora de editar, aparece el último que haya sido editado.

```
> SL AV 10 BL AV 5
> GD 90 AV 60 GD 90
> AV 5 RE 5 GD 90
> AV 15 GD 90 AV 55
> GD 90 AV 5 GI 180
> AV 40 GD 90 AV 1
> GI 90 RE 40 AV 2
> GD 90 AV 3 GI 90
> AV 35 RE 33 GD 90
> AV 3 GI 90 RE 2
> AV 35 RE 35 GD 90
> AV3 GI 90 AV 35
> RE 35 GD 180 AV 2
> GI 45 AV 3 GI 80
> AV 37
> SL CENTRO
> RE 45 GI 90 AV 15
> GD 90 BL AV 55
> GI 90 AV 2 GD 90
> AV 4 GD 90 AV 2
> GI 90 AV 3 GI 90
> AV 2 GD 90 AV 3
> GI 90 AV 2 GD 45
> AV 3 GD 80 AV 37
> SL GI 30 RE 20
> GI 90 AV 10 GD 135
> BL
> REPITE 7 [AV 4 GD 14]
> REPITE 4 [RE 9 GI 40]
> FIN
```

4:

Primero hacemos el recuadro:

```
? PARA RECUADRO
> SL AV 60
> GD 90 BL
> REPITE 2 [AV 120 GD 90 AV 200 GD 90]
> FIN
```

Definimos los ejes:

```
? PARA EJES
> SL CENTRO
> BL AV 45
> RE 90
> CENTRO
> GD 90 AV 75
> RE 150
> FIN
```

A continuación la escala horizontal:

```
? PARA ESHO
> AV 5 GI 90
> RE 3
> REPITE 15 [BL AV 6 RE 6 GD 90 SL AV 10 GI 90]
> FIN
```

Y ahora la escala vertical:

```
? PARA ESVE
> SL CENTRO
> AV 40 GI 90
> RE 3
> REPITE 9 [BL AV 6 RE 6 GI 90 SL AV 10 GD 90]
> FIN
```

Ya tenemos definidos los cuatro procedimientos por separados. Observa que tomamos en cuenta el lugar donde está la tortuga cuando acaba cada uno de los procedimientos para que cuando empiece a dibujar el siguiente lo haga correctamente.

Ahora sólo nos queda hacer uno general que contenga los cuatro anteriores. En éste vendrá la inicialización tanto del estado de la pantalla como de la tortuga:

```
? PARA GRAFICO
> PM
> BP
> OT
> RECUADRO
> EJES
> ESHO
> ESVE
> FIN
```

Si ahora introduces:

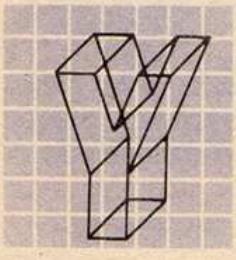
```
? GRAFICO
```

Obtendremos el dibujo pedido.

5:

- PARA PARA: INCORRECTA. No se puede dar el nombre de una primitiva a un procedimiento.
- EDITA "FF: CORRECTA.
- ED CASA: INCORRECTA. Falta poner las comillas delante del nombre del procedimiento.
- GUARDA "LAV: CORRECTA.
- GUARDA "REPITE: CORRECTA. Aunque REPITE es una primitiva, se está utilizando en este caso como un nombre de fichero, no de un procedimiento.
- PARA LIBRO: CORRECTA.

La zona de memoria donde se almacenan los procedimientos se llama área de trabajo.



A hemos visto en anteriores tomos cómo realizar el movimiento vertical y horizontal de figuras complejas. Aunque nos falta todavía por ver el movimiento diagonal y circular de este tipo de figuras, vamos a desarrollar durante todo este tomo el programa que se propuso en el anterior. En el tomo 6 se dieron los dibujos de un hombrecillo en tres posiciones diferentes. Se pedía que hicieseis un programa que moviese un hombrecillo como ese por la pantalla de arriba hacia abajo, utilizando los conceptos que hemos aprendido hasta ahora.

Como este es un programa un poco más complicado que los anteriores y como es la primera vez que vamos a utilizar caracteres definibles en uno de nuestros programas, gastaremos el contenido de todo este tomo explicando cómo puede hacerse este programa.

En el tomo anterior os dimos las posiciones que tendría que tener la figura del hombrecillo. Ahora os las volvemos a repetir, para

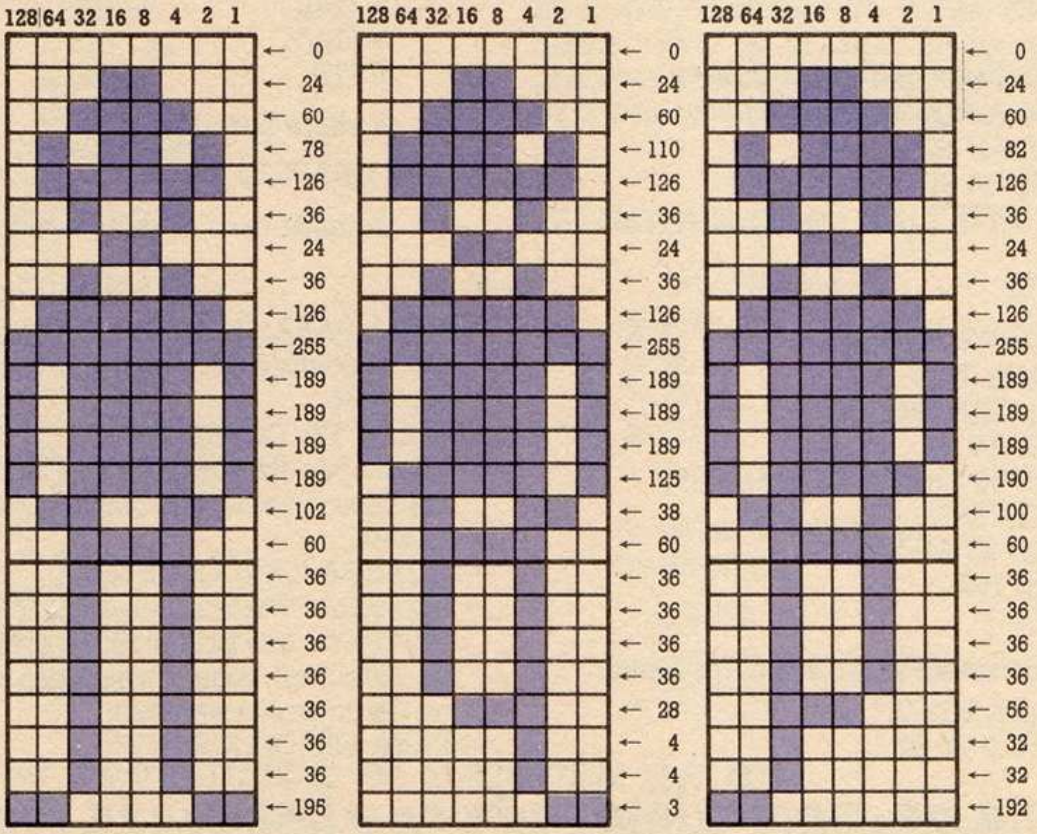


Fig. 1. Estas son las figuras con las definiciones de los tres movimientos de nuestro hombrecillo. También se incluyen los valores de todos los bytes que las componen.

que no tengáis que estar mirando constantemente el tomo anterior, en la figura 1.

Como podéis ver, cada figura está compuesta de tres caracteres, uno debajo de otro. Como tenemos tres figuras, esto significa que tenemos que definir nueve caracteres distintos. En uno de los primeros tomos vimos cómo podíamos hacerlo. En éste veremos cómo definir estos nueve caracteres. Si no te gusta cómo hemos diseñado el hombrecillo, puedes definirte tu propio personaje y utilizarlo en vez del nuestro.

Veamos ahora cómo definir este muñeco en los distintos ordenadores.

Para Spectrum

Para definir estos nueve caracteres utilizaremos los caracteres definibles por el usuario (UDG), que empiezan en el carácter 144. Como tenemos que definir nueve caracteres utilizaremos los que tienen un código ASCII comprendido entre el 144 y 152, ambos inclusive.

El programa que nos define dichos nueve caracteres es el programa 1. Este programa entrará a formar parte del programa general del movimiento del hombrecillo en su versión para SPECTRUM.

Como puede apreciarse, el programa va leyendo una serie de valores de una serie de líneas con DATA. En estas líneas se encuentran las definiciones de los nueve caracteres, agrupados de tres en tres. Esta agrupación es sólo para que se vea dónde empieza un hombrecillo y dónde empieza el otro, pero se podría haber puesto todo en la misma línea.

El uso de la función USR, como ya se explicó en tomos anteriores, sirve para introducir en memoria las nuevas definiciones de los caracteres. No olvides que para visualizar dichos caracteres una vez definidos podemos hacer dos cosas:

1. Decirle al ordenador que imprima dicho carácter dándole su código ASCII y utilizando la función CHR$.

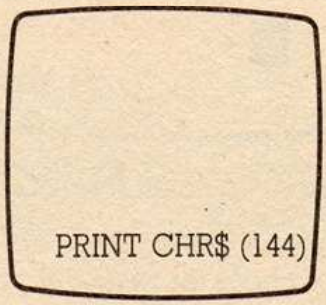


Fig. 2. Si escribimos en el Spectrum "PRINT CHR$(144)" antes de definir dicho carácter nos aparecerá la letra 'A'. Igual ocurriría con el resto de los caracteres definibles por el usuario (UDG).

2. Poniendo el cursor en modo GRAPHICS y pulsando la tecla que le corresponda. Para poner el cursor en modo GRAPHICS sólo tienes que pulsar la tecla CAPS SHIFT y el 9 a la vez. Una vez hecho esto el cursor aparecerá como una G parpadeante. En este momento puedes pulsar cualquier tecla entre la A y la U, con lo que aparecerá dicho carácter en pantalla. Si el carácter no ha sido definido aún, entonces aparecerá la tecla que has pulsado.

```
 40 REM
 50 REM *** DEFINICION DEL HOMBRECILLO EN EL SPECTRUM ***
 60 REM
 70 FOR I=USR "A" TO USR "J"-1
 80    READ A
 90    POKE I,A
100 NEXT I
110 REM
120 REM *** DATAS QUE DEFINEN EL HOMBRECILLO ***
130 REM
140 REM *** POSICION CENTRAL ***
150 REM
160 DATA 0,24,60,78,126,36,24,36
170 DATA 126,255,189,189,189,189,102,60
180 DATA 36,36,36,36,36,36,36,195
190 REM
200 REM *** PIERNA DERECHA LEVANTADA ***
210 REM
220 DATA 0,24,60,110,126,36,24,36
230 DATA 126,255,189,189,189,125,38,60
240 DATA 36,36,36,36,28,4,4,3
250 REM
260 REM *** PIERNA IZQUIERDA LEVANTADA ***
270 REM
280 DATA 0,24,60,82,126,36,24,36
290 DATA 126,255,189,189,189,190,100,60
300 DATA 36,36,36,36,56,32,32,192
```

Programa 1.

Para volver a poner el cursor en el modo normal tienes que volver a pulsar a la vez las teclas CAPS SHIFT y 9. Como dijimos en anteriores tomos, cuando en un programa aparezca una letra subrayada, esto significará que tienes que poner el cursor en modo GRAPHICS antes de meterla en tu ordenador.

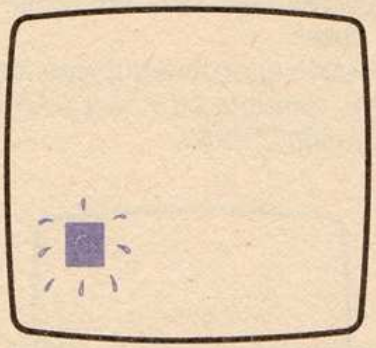


Fig. 3. Cuando pulsamos a la vez las teclas CAPS SHIFT y 9 el cursor se convierte en una G parpadeante que nos indica que estamos en el modo GRAPHIS.

Por lo demás, el resto del programa no reviste ninguna complicación, por lo que si quieres ver cómo se mueve tu hombrecillo por la pantalla, sólo tienes que introducir el programa general y realizar las modificaciones que estén indicadas para el SPECTRUM.

```
10 CLS
20 PRINT AT X, Y; «ABC»
40 GOTO 3247
100 LET A = SIN(x) * AV
110 LET B = 100 * A
120 FOR J = TR TO XR
130 GOSUB 1000
140 PRINT «M»
150 GOSUB 330
```

Fig. 4. Cuando en un programa aparezcan una o más letras entre comillas, significa que tenemos que pulsar CAPS SHIFT y 9 a la vez, para pasar al modo GRAPHIS, después pulsar la tecla que está subrayada.

Para ver los caracteres que hemos definido, te recomiendo que introduzcas el programa 2. Si alguno de los caracteres no aparece en la pantalla como debiera, repasa las líneas DATA's del programa hasta encontrar el error.

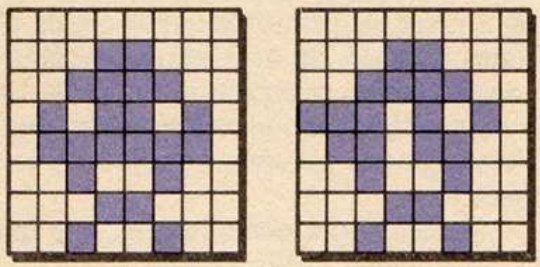


Fig. 5. Si al imprimirse los caracteres en la pantalla aparece alguno que esté mal, repasa las DATA, porque te habrás equivocado.

```
1000 CLS
1010 LET K=1
1020 FOR I=144 TO 152 STEP 3
1030     FOR J=0 TO 2
1040         PRINT AT J,K;CHR$(I+J)
1050     NEXT J
1060     LET K=K+3
1070 NEXT I
1080 STOP
```

Programa 2.

Para MSX

En el MSX, para definir los nueve caracteres que componen las tres posiciones distintas del movimiento del hombrecillo, tenemos que utilizar los SPRITES. Como tenemos nueve caracteres distintos, lógicamente, tendremos que utilizar nueve SPRITES distintos.

El programa que prepara al ordenador para que tenga definidos estos nueve SPRITES es el 3. Este programa forma parte del programa general del movimiento del hombrecillo.

Como ya vimos en un tomo anterior, para definir SPRITES en el MSX es necesario utilizar la función SRITE$ (n) = string. Donde n

```
 40 REM
 50 REM *** DEFINICION DEL HOMBRECILLO EN EL MSX ***
 60 REM
 70 FOR I=1 TO 9
 80     A$=""
 90     FOR J=1 TO 8
100         READ A
110         A$=A$+CHR$(A)
120     NEXT J
130     SPRITE$(I)=A$
140 NEXT I
150 REM
160 REM *** DATAS QUE DEFINEN EL HOMBRECILLO ***
```

```
170 REM
180 REM *** POSICION CENTRAL ***
190 REM
200 DATA 0,24,60,78,126,36,24,36
210 DATA 126,255,189,189,189,189,102,60
220 DATA 36,36,36,36,36,36,36,195
230 REM
240 REM *** PIERNA DERECHA LEVANTADA ***
250 REM
260 DATA 0,24,60,110,126,36,24,36
270 DATA 126,255,189,189,189,125,38,60
280 DATA 36,36,36,36,28,4,4,3
290 REM
300 REM *** PIERNA IZQUIERDA LEVANTADA ***
310 REM
320 DATA 0,24,60,82,126,36,24,36
330 DATA 126,255,189,189,189,190,100,60
340 DATA 36,36,36,36,56,32,32,192
```

Programa 3.

es el número de SPRITE y string es una cadena alfanumérica con ocho caracteres de longitud. Como el SPRITE está definido como una cuadrícula de 8 × 8 puntos y como cada BYTE tiene 8 BITS, necesitamos ocho BYTES para poder definir un SPRITE.

Según esto, cada carácter del string que se encuentra en la definición del SPRITE tiene un código ASCII, cuyo número corresponde con el valor de cada línea del SPRITE.

El resto del programa no reviste ninguna dificultad. Todo consiste en dos bucles anidados (uno dentro de otro). El primero de ellos va seleccionando el SPRITE que vamos a definir. El segundo va tomando los valores de las líneas de DATA uno a uno y los va almacenando dentro de una variable alfanumérica (STRING). Cuando termina este segundo bucle se asigna el string al SPRITE correspondiente y se continúa con el siguiente hasta que se han definido los nueve SPRITES que necesitábamos.

Para ver si los nueve SPRITES han salido bien y no nos hemos equivocado al copiar las líneas del data, os propongo el programa 4. Cada vez que pulses una tecla aparecerá un SPRITE distinto en la pantalla. Si alguno de ellos no aparece con la forma que debiera, repásate las líneas de DATAS para remediar el error.

```
1000 SCREEN 2
1010 FOR I=1 TO 6
1020     PUT SPRITE I,(100,100),15,I
1030     A$=INPUT$(1)
1040     PUT SPRITE 0,(100,100),,I
1050 NEXT I
1060 END
```

Programa 4.

Cuando ejecutes el programa 4, para el siguiente sólo tienes que pulsar una tecla.

Este programa tienes que unirlo con el programa general de movimiento del hombrecillo y realizar las modificaciones que se te indiquen para su funcionamiento.

Para AMSTRAD

Si os acordáis del tomo 2, en él explicábamos cómo definir caracteres. Ya vimos que podíamos definir todos los caracteres que tiene el AMSTRAD. Según esto, no podemos tener ningún problema para definir los nueve caracteres que componen las tres figuras del hombrecillo en movimiento. El programa 5 es parte del programa general que generará dicho movimiento, y su función es definir dichos caracteres.

Para definir nuestros propios caracteres en el AMSTRAD utilizamos la función

```
 40 REM
 50 REM *** DEFINICION DEL HOMBRECILLO EN EL AMSTRAD ***
 60 REM
 70 FOR I=1 TO 9
 80     READ A,B,C,D,E,F,G,H
 90     SYMBOL 239+I,A,B,C,D,E,F,G,H
100 NEXT I
110 REM
120 REM *** DATAS QUE DEFINEN EL HOMBRECILLO ***
130 REM
140 REM *** POSICION CENTRAL ***
150 REM
160 DATA 0,24,60,78,126,36,24,36
170 DATA 126,255,189,189,189,189,102,60
```

```
180 DATA 36,36,36,36,36,36,36,195
190 REM
200 REM *** PIERNA DERECHA LEVANTADA ***
210 REM
220 DATA 0,24,60,110,126,36,24,36
230 DATA 126,255,189,189,189,125,38,60
240 DATA 36,36,36,36,28,4,4,3
250 REM
260 REM *** PIERNA IZQUIERDA LEVANTADA ***
270 REM
280 DATA 0,24,60,82,126,36,24,36
290 DATA 126,255,189,189,189,190,100,60
300 DATA 36,36,36,36,56,32,32,192
```

Programa 5.

SYMBOL. Con ella podemos definir todos los caracteres comprendidos entre el 240 y el 255, ambos inclusive. Si queremos poder definir otros caracteres con código ASCII menor que 240, tendremos que utilizar la función:

SYMBOL AFTER n

donde n es el último carácter protegido de nuestra definición. O lo que es igual, n+1 es el primer carácter que podemos definir.

Como recordaréis, la sintaxis de la sentencia SYMBOL es la siguiente:

SYMBOL n, L1, L2, L3, L4, L5, L6, L7, L8

donde n es el número de carácter que queremos definir y L1, L2, L3,... L8 son las ocho líneas que definen dicho carácter.

Por lo demás, el programa no tiene ninguna dificultad. Lo único que hace es leer grupos de ocho BYTES a la vez para definir los nuevos caracteres. Esto se realiza mediante un bucle de 1 a 9, ya que nueve son los caracteres necesarios para definir las tres posiciones distintas del muñeco.

Para ver si los caracteres están bien definidos, te propongo ejecutar el programa 6. Si ves que alguno tiene una forma rara, repasa las líneas de DATA del programa 5 hasta encontrar algún error.

```
1000 CLS
1010 LET K=1
1020 FOR I=240 TO 248 STEP 3
1030     FOR J=0 TO 2
1040         LOCATE J,K
1050         PRINT CHR$(I+J)
1060     NEXT J
1070     LET K=K+3
1080 NEXT I
1090 END
```

Programa 6.

Este programa forma parte del programa general, que es el que se encarga del movimiento del hombrecillo. Por lo que tendrás que unirlo para que este último pueda funcionar. También tienes que fijarte en las modificaciones que son necesarias realizar en el AMSTRAD para que este programa funcione.

Para COMMODORE

Definir caracteres en el Commodore resulta un poco difícil, por eso lo que haremos será definir una serie de SPRITES. Como los SPRITES en el COMMODORE no están formados por una retícula de 8 × 8 puntos, sino por una de 24 × 21 puntos, esto lo utilizaremos para no tener que definir nueve SPRITES, sino sólo seis. Esto es así porque cada muñequito cabe en dos SPRITES y no en tres caracteres.

La figura 6 nos muestra cómo quedarían dichos SPRITES una vez modificados. Como el COMMODORE tiene los SPRITES más grandes que el resto de los ordenadores, podrías aprovechar esta propiedad para realizar la figura de un muñeco más complicado.

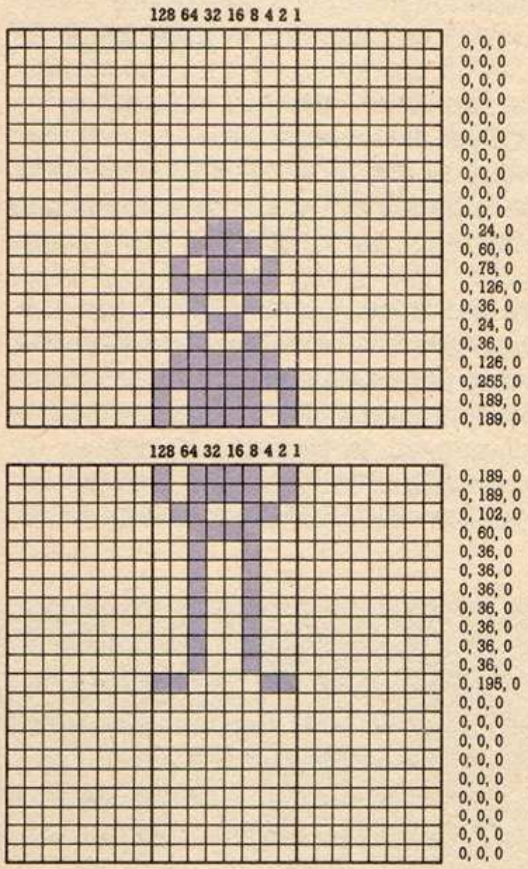


Fig. 6a. Definición del muñequito de frente en el Commodore con todos los valores de los bytes.

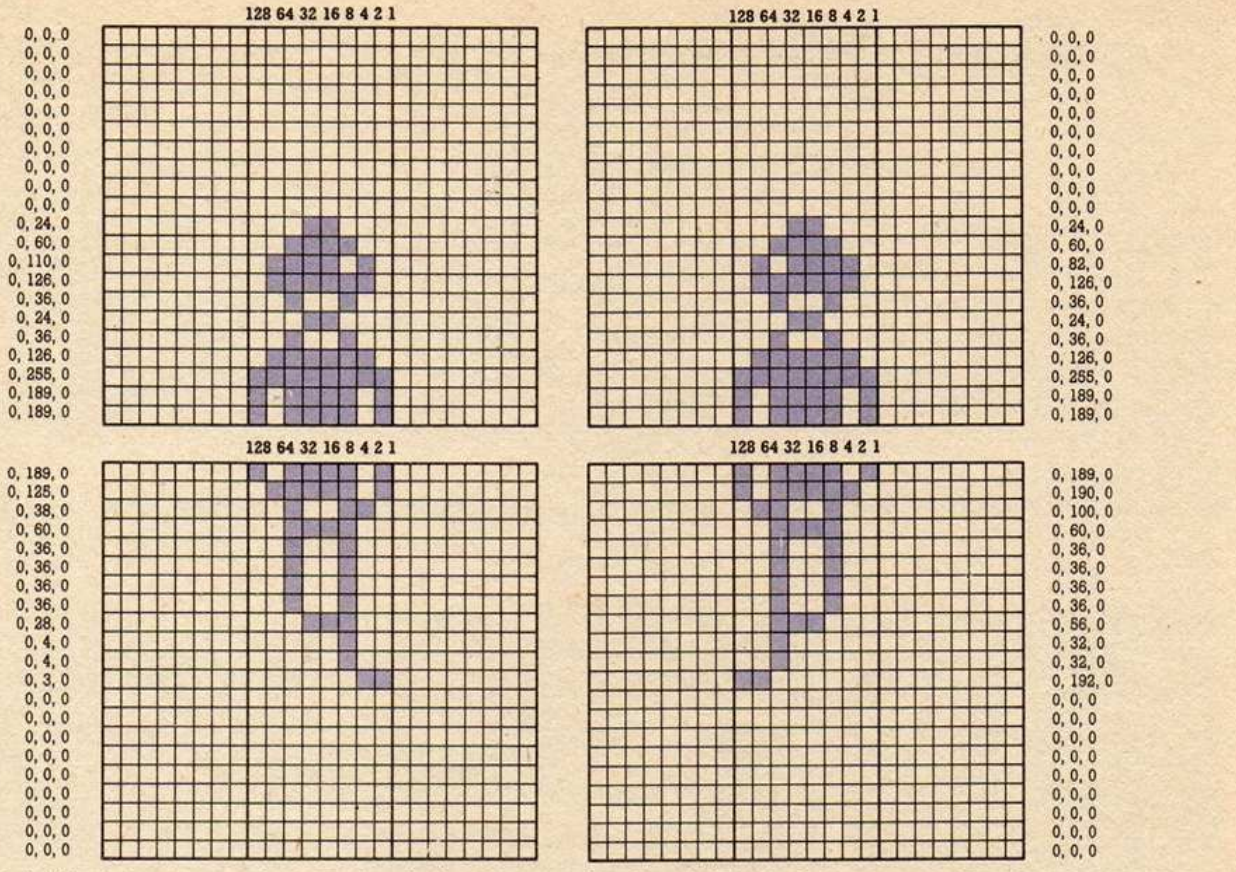


Fig. 6b. Definición del muñeco con la pierna derecha levantada.

Fig. 6c. Definición del muñeco con la pierna izquierda levantada.

El programa 7 sirve para definir dichos SPRITES. Este programa forma parte del general y tendrás que unirlo a éste cuando lo teclees.

```
 40 REM
 50 REM *** DEFINICION DEL HOMBRECILLO EN EL COMMODORE ***
 60 REM
 70 V=53248
 80 FOR I=1 TO 6
 90    POKE 2039+I,191+I
100 NEXT I
110 FOR I=0 TO 377
120    READ M
130    POKE 12288+I,N
140 NEXT I
150 REM
160 REM *** DATAS QUE DEFINEN EL HOMBRECILLO ***
170 REM
180 REM *** POSICION CENTRAL ***
190 REM --- SPRITE SUPERIOR ---
200 REM
210 DATA 0,0,0,0,0,0,0,0,0
220 DATA 0,0,0,0,0,0,0,0,0
230 DATA 0,0,0,0,0,0,0,0,0
240 DATA 0,0,0,0,24,0,0,60,0
250 DATA 0,78,0,0,126,0,0,36,0
260 DATA 0,24,0,0,36,0,0,126,0
270 DATA 0,255,0,0,189,0,0,189
280 REM
290 REM --- SPRITE INFERIOR ---
300 REM
310 DATA 0,189,0,0,189,0,0,102,0
320 DATA 0,60,0,0,36,0,0,36,0
330 DATA 0,36,0,0,36,0,0,36,0
340 DATA 0,36,0,0,36,0,0,195,0
350 DATA 0,0,0,0,0,0,0,0,0
360 DATA 0,0,0,0,0,0,0,0,0
370 DATA 0,0,0,0,0,0,0,0,0
380 REM
390 REM *** PIERNA DERECHA LEVANTADA ***
```

```
400 REM
410 REM --- SPRITE SUPERIOR ---
420 REM
430 DATA 0,0,0,0,0,0,0,0,0
440 DATA 0,0,0,0,0,0,0,0,0
450 DATA 0,0,0,0,0,0,0,0,0
460 DATA 0,0,0,0,24,0,0,60,0
470 DATA 0,110,0,0,126,0,0,36,0
480 DATA 0,24,0,0,36,0,0,126,0
490 DATA 0,255,0,0,189,0,0,189,0
500 REM
510 REM --- SPRITE INFERIOR ---
520 REM
530 DATA 0,189,0,0,125,0,0,38,0
540 DATA 0,60,0,0,36,0,0,36,0
550 DATA 0,36,0,0,36,0,0,28,0
560 DATA 0,4,0,0,4,0,0,3,0
570 DATA 0,0,0,0,0,0,0,0,0
580 DATA 0,0,0,0,0,0,0,0,0
590 DATA 0,0,0,0,0,0,0,0,0
600 REM
610 REM *** PIERNA IZQUIERDA LEVANTADA ***
620 REM
630 REM --- SPRITE SUPERIOR ---
640 REM
650 DATA 0,0,0,0,0,0,0,0,0
660 DATA 0,0,0,0,0,0,0,0,0
670 DATA 0,0,0,0,0,0,0,0,0
680 DATA 0,0,0,0,24,0,0,60,0
690 DATA 0,82,0,0,126,0,0,36,0
700 DATA 0,24,0,0,36,0,0,126,0
710 DATA 0,255,0,0,189,0,0,189,0
720 REM
730 REM --- SPRITE INFERIOR ---
740 REM
750 DATA 0,189,0,0,190,0,0,100,0
760 DATA 0,60,0,0,36,0,0,36,0
770 DATA 0,36,0,0,36,0,0,56,0
780 DATA 0,32,0,0,32,0,0,192,0
790 DATA 0,0,0,0,0,0,0,0,0
800 DATA 0,0,0,0,0,0,0,0,0
810 DATA 0,0,0,0,0,0,0,0,0
```

Programa 7.

Como puedes ver en este programa, y como ya explicamos en el tomo 2, cada SPRITE está contenido en un área de memoria distinta. Como tenemos seis SPRITES distintos, lo más normal es que lo localicemos en zonas contiguas de memoria. Estas zonas son la 192 para el SPRITE 1, la 193 para el 2, la 194 para el 3, y así sucesivamente hasta la zona 197 para el SPRITE 6. Para decirle al ordenador en qué zona va a estar la definición de cada SPRITE, pokeamos el número de la zona a partir de la posición 2040. Así, para decir dónde está el SPRITE 1, pokeamos el número 192 en la dirección 2040; para decir dónde se encuentra el 2, pokeamos el número 193 en la posición 2041. Esto se realiza seis veces, hasta pokear el número 197 en la dirección de memoria 2045.

Una vez que le hemos dicho al ordenador en qué zonas de memoria van a ir alojados todos los SPRITES, comenzamos un bucle desde 0 a 377 (el número de BYTES que definen los seis SPRITES) y pokeamos los números que se encuentran almacenados en las sentencias DATA a partir de la dirección de memoria 12288. Esta es la dirección donde empieza la zona 192. Esta zona, que ocupa 64 BYTES, está justamente antes de la zona 193. Por ello, el bucle es continuo y pokeamos los valores de las sentencias DATA uno detrás de otros, pues después de una zona viene la siguiente. Por esta razón, hemos elegido zonas continuas para la definición de los SPRITES, ya que si hubiésemos elegido zonas que no lo fuesen, hubiésemos tenido que realizar más de un bucle para poner los datos de cada SPRITE en su zona.

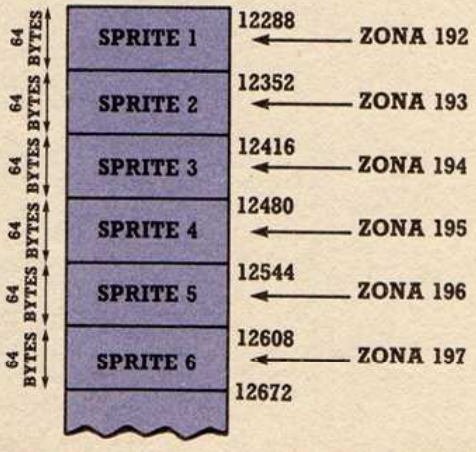


Fig. 7. Las zonas de memoria en el Commodore son contiguas.

Una vez que hayas introducido y tecleado este programa, si quieres ver cómo te ha quedado la definición de los SPRITES te propongo que introduzcas el programa 8 para que te aparezcan en la pantalla.

```
1000 PRINT "<SHIFT-HOME>"
1010 LET K=10
1020 LET V=53248
1030 POKE V+21,63
1040 FOR I=0 TO 12 STEP 2
1050    POKE V+I,K
1060    POKE V+I+1,100
1070 NEXT I
1080 END
```

Programa 8.

Si ves que alguno de los SPRITES no aparece como debiera, busca en las líneas DATA el error, pues seguro que te has equivocado al transcribir el programa.

Para IBM

Como se vio en el tomo 2, en el IBM no pueden definirse caracteres. También dijimos que eso no era un problema sin solución, pues había otras formas de hacerlo, aunque no dijimos cómo. En este tomo todavía no estamos preparados para ver cómo podemos hacerlo, pero a continuación os propongo el programa completo, no sólo la parte de definición del hombrecillo, para que veáis que se puede realizar.

```
10 REM ********************************
20 REM * MOVIMIENTO DE UN MUÑECO      *
30 REM * POR LA PANTALLA DE ARRIBA    *
40 REM * HACIA ABAJO                  *
50 REM *                             *
60 REM * VERSION VALIDA PARA IBM      *
70 REM ********************************
80 REM
100 DIM A%(30),B%(30),C%(30)
110 SCREEN 1:CLS
120 GET (133,39)-(140,61),A%
130 GET (146,39)-(153,61),B%
140 GET (162,39)-(169,61),C%
150 FOR I=1 TO 30
160    READ A%(I),B%(I),C%(I)
170 NEXT I
180 CLS
190 SW=2
200 FOR I=1 TO 180 STEP 2
210    ON SW GOSUB 1000,2000,3000,4000
220    FOR J=1 TO 200
230    NEXT J
240 NEXT I
250 END
1000 REM
1010 REM *** PIERNA DERECHA ***
1020 REM
1030 PUT (100,I),B%,PSET
1040 SW=2
1050 RETURN
2000 REM
2010 REM *** POSICION CENTRAL ***
2020 REM
2030 GOSUB 5000
2040 SW=3
2050 RETURN
3000 REM
3010 REM *** PIERNA IZQUIERDA ***
3020 REM
3030 PUT (100,I),C%,PSET
3040 SW=4
3050 RETURN
4000 REM
4010 REM *** POSICION CENTRAL ***
4020 REM
4030 GOSUB 5000
4040 SW=1
4050 RETURN
5000 REM
5010 REM *** IMPRESION DE LA P. CENTRAL ***
5020 REM
5030 PUT (100,I),A%,PSET
5040 RETURN
7000 DATA 27,27,27,0,0,0,0,0,0,16385,16385,16385,20485,20485,20485
7010 DATA 17425,17429,21521,21525,21525,21525,4100,4100,4100
```

```
7020 DATA 16385,16385,16385,4100,4100,4100,21525,21525,21525
7030 DATA 21845,21845,21845,20805,20805,20805,20805,20805,20805
7040 DATA 20805,20805,20805,20805,20757,21573,5140,5124,4116
7050 DATA 20485,20485,20485,4100,4100,4100,4100,4100,4100
7060 DATA 4100,4100,4100,4100,4100,4100,4100,4096,16389
7070 DATA 4100,4096,4,4100,4096,4,1360,1280,80,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

Programa 9.

Como todavía no estamos preparados, dejaremos la explicación del funcionamiento de este programa para sucesivos tomos.

Programa de movimiento de un hombre

Ya ha llegado el momento de que veamos a nuestro personaje moviéndose por la pantalla. El programa que lo realiza es el 10.

```
  10 REM ****************************************
  15 REM * MOVIMIENTO DE UN HOMBRECILLO         *
  20 REM * DE ARRIBA A ABAJO POR LA PANTALLA    *
  25 REM ****************************************
  30 REM
1000 REM
1010 REM *** MOVIMIENTO DEL HOMBRECILLO ***
1020 REM
1030 CLS
1040 LET SW=1
1050 FOR I=2 TO 18
1060     IF SW=0 THEN GOSUB 2000
1070     IF SW=1 THEN GOSUB 3000
1080     IF SW=2 THEN GOSUB 4000
1090     IF SW=3 THEN GOSUB 5000
1100     FOR J=1 TO 100
1110     NEXT I
1120 NEXT I
1130 END
2000 REM
2010 REM *** PIERNA DERECHA LEVANTADA ***
2020 REM
2030 LOCATE I,10:PRINT " "
2040 LOCATE I+1,10:PRINT CHR$(243)
2050 LOCATE I+2,10:PRINT CHR$(244)
2060 LOCATE I+3,10:PRINT CHR$(245)
2070 LET SW=1
2080 RETURN
3000 REM
3010 REM *** POSICION CENTRAL ***
3020 REM
3030 GOSUB 6000
3040 LET SW=2
3050 RETURN
4000 REM
4010 REM *** PIERNA IZQUIERDA LEVANTADA ***
4020 REM
4030 LOCATE I,10:PRINT " "
4040 LOCATE I+1,10:PRINT CHR$(246)
4050 LOCATE I+2,10:PRINT CHR$(247)
4060 LOCATE I+3,10:PRINT CHR$(248)
4070 LET SW=3
4080 RETURN
5000 REM
5010 REM *** POSICION CENTRAL ***
5020 REM
5030 GOSUB 6000
5040 LET SW=0
5050 RETURN
6000 REM
6010 REM *** IMPRESION EN LA POSICION CENTRAL ***
6020 REM
6030 LOCATE I,10:PRINT " "
6040 LOCATE I+1,10:PRINT CHR$(240)
6050 LOCATE I+2,10:PRINT CHR$(241)
6060 LOCATE I+3,10:PRINT CHR$(242)
6070 RETURN
```

Programa 10.

Este programa funciona perfectamente en el AMSTRAD; para los demás ordenadores las modificaciones son las siguientes:

Commodore

```
1030 PRINT "<SHIF-HOME>"
1040 LET SW = 1;LET V = 53248
1050 FOR I = 10 TO 200 STEP 4
2030 POKE V + 21, 12
2040 POKE V + 4, 100: POKE V + 6, 100
2050 POKE V + 5, I: POKE V + 7, I + 21
2060 REM
4030 POKE V + 21, 48
4040 POKE V + 8, 100: POKE V + 10, 100
4050 POKE V + 9, I: POKE V + 11, I + 21
4060 REM
6030 POKE V + 21, 3
6040 POKE V + 0, 100: POKE V + 2, 100
6050 POKE V + 1, I: POKE V + 3, I + 21
6060 REM
```

MSX

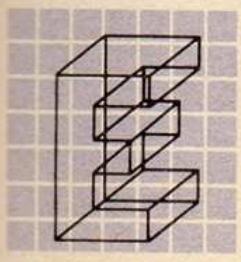
```
1030 SCREEN 2
1050 FOR I = 10 TO 150 STEP 4
2030 PUT SPRITE 0, (100, I), 15, 4
2040 PUT SPRITE 1, (100, I + 8), 15, 5
2050 PUT SPRITE 2, (100, I +16), 15, 6
2060 REM
4030 PUT SPRITE 0, (100, I), 15, 7
4040 PUT SPRITE 1, (100, I + 8), 15, 8
4050 PUT SPRITE 2, (100, I + 16), 15, 9
4060 REM
6030 PUT SPRITE 0, (100, I), 15, 1
6040 PUT SPRITE 1, (100, I + 8), 15, 2
6050 PUT SPRITE 2, (100, I + 16), 15, 3
```

Spectrum

```
2030 PRINT AT I, 10; " "
2040 PRINT AT I + 1, 10; "D"
2050 PRINT AT I + 2, 10; "E"
2060 PRINT AT I + 3, 10; "F"
4030 PRINT AT I, 10; " "
4040 PRINT AT I + 1, 10; "G"
4050 PRINT AT I + 2, 10; "H"
4060 PRINT AT I + 3, 10; "I"
6030 PRINT AT I, 10; " "
6040 PRINT AT I + 1, 10; "A"
6050 PRINT AT I + 2, 10; "B"
6060 PRINT AT I + 3, 10; "C"
```

Como podéis apreciar al observar el programa, éste no es ninguna maravilla. Se puede mejorar muchísimo. Se ha hecho de esta manera para que las modificaciones para los distintos ordenadores sean más sencillas y para que tengáis que teclear menos tiempo. Como ejercicio os proponemos una versión de este programa para vuestro ordenador.

Trucos de programación



N el tomo anterior vimos lo importante que es saber utilizar y crear ficheros para cualquier aplicación que nosotros necesitemos hacer. También planteamos un programa gestor de ficheros. En este fascículo vamos a meternos más en el tema y vamos a ver algunas de las rutinas que componen dicho programa.

Lo único que nos quedó por ver fue la elección de los delimitadores o separadores de campos. No se dijo si se utilizaría un solo delimitador o, por el contrario, utilizaríamos uno para cada tipo de campo. La mejor solución es utilizar uno sólo como delimitador de todos los campos. Este delimitador tiene que ser un caracter que no pueda utilizar el usuario, pues si no dividiría cada ficha en más campos de los permitidos. Este carácter ha de ser, entonces, uno cuyo código ASCII sea bastante alto, como, por ejemplo, el carácter 254.

No importa qué forma tenga este carácter, lo único que importa es saber que actúa como delimitador o deparador de los diferentes campos de un registro.

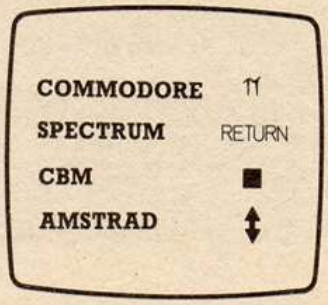


Fig. 1. Estos son los caracteres en código ASCII 254 en los distintos ordenadores.

Una vez visto cuál va a ser el carácter separador empezamos a realizar las rutinas que vamos a ir necesitando. Algunas de estas rutinas ya han sido publicadas en esta colección, pero las volvemos a poner aquí porque tenemos que cambiarles los números de línea. Estas rutinas son:

— Rutina de entrada de datos de propósito general.

— Rutina de recogida de opción en el menú.

— Rutina de pulsa una tecla.

y su programa se encuentra a continuación, pero sin tantas líneas REM como en la versión que dimos en tomos anteriores. La explicación del funcionamiento de estas rutinas la puedes encontrar en el tomo en que aparecen.

```
8000 REM
8001 REM ***********************************
8002 REM * SUBRUTINA DE ENTRADA DE DATOS *
8003 REM ***********************************
8004 REM
8005 LET D$="" : LET LO=0
8006 LOCATE Y,X
8007 FOR Z=1 TO LO
8008     PRINT ".";
8009 NEXT Z
```

```
8010 LOCATE Y,X
8011 PRINT "_"
8012 LOCATE Y,X
8013 LET A$=INKEY$
8014 IF A$="" THEN GOTO 8013
8015 IF A$=CHR$(8) AND LO>0 THEN LET LO=LO-1:LET D$=LEFT$(D$,LO) :LET X=X-1:LOCA
TE Y,X:PRINT "_.":LOCATE Y,X:GOTO 8013
8016 IF A$=" " OR A$="." OR A$="-" THEN GOTO 8019
8017 IF A$=CHR$(13) THEN GOTO 8025
8018 IF A$>M$ OR A$<W$ THEN GOTO 8013
8019 PRINT A$;"_"
8020 LET X=X+1
8021 LOCATE Y,X
8022 LET D$=D$+A$
8023 LET LO=LO+1
8024 IF LO<>LO THEN GOTO 8013
8025 LOCATE Y,X
8026 FOR Z=LO TO LO
8027    PRINT " ";
8028 NEXT Z
8029 RETURN
```

Programa 1.

Las modificaciones que hay que hacer para que este programa funcione en ordenadores distintos del IBM y del AMSTRAD son las siguientes:

COMMODORE:

8006 POKE 214,Y:POKE 211,X
8010 POKE 214,Y:POKE 211,X
8012 POKE 214,Y:POKE 211,X
8013 GET A$
8015 IF A$=CHR$(20) AND L0>0 THEN L0=L0-1:D$=LEFT$(D$,L0)
:X=X-1:POKE 214,Y:POKE 211,X:PRINT "_":POKE 214,Y
:POKE 211,X:GOTO 8013
8021 POKE 214,Y:POKE 211,X
8025 POKE 214,Y:POKE 211,X

MSX:

8006 LOCATE X,Y
8010 LOCATE X,Y
8012 LOCATE X,Y
8015 IF A$=CHR$(8) AND L0>0 THEN L0=L0-1
:D$=LEFT$(D$,L0):X=X-1:LOCATE X,Y:PRINT "_"
:LOCATE X,Y:GOTO 8013
8021 LOCATE X,Y
8025 LOCATE X,Y

SPECTRUM:

8006 PRINT AT Y,X;
8010 PRINT AT Y,X;
8012 PRINT AT Y,X;
8015 IF A$=CHR$(8) AND L0>0 THEN LET L0=L0-1 :LET D$=D$ (TO L0): LET X=X-1: PRINT AT Y,X;" "; AT Y,X;:GOTO 8013
8021 PRINT AT Y,X;
8025 PRINT AT Y,X;

Fig. 2. Utilizaremos el programa 1 para realizar la entrada de datos de las fichas.

A continuación aparece el programa 2. Esta es una rutina de pulsar una tecla. Se utilizará en la recogida de datos en los menús.

Las modificaciones que hay que hacer para que este programa funcione en ordenadores distintos del AMSTRAD y del IBM son las que aparecen a continuación:

```
8100 REM
8101 REM ********************************
8102 REM * MENSAJE Y RECOGIDA DE OPCION *
8103 REM ********************************
8104 REM
8105 LET T$="INTRODUZCA OPCION"
8106 LET L=17
8107 LOCATE Y,X
8108 PRINT T$;
8109    LOCATE Y,X+19
8110    PRINT CHR$(177);
8111    LET A$=INKEY$
8112    IF A$<>"" AND (A$<W$ OR A$>M$) THEN GOSUB 8127
8113    IF B$<>"" THEN GOSUB 8138:GOTO 8124
8114    FOR J=1 TO 100
8115    NEXT J
8116    LOCATE Y,X+19
8117    PRINT " ";
8118    LET A$=INKEY$
8119    IF A$<>"" AND (A$<W$ OR A$>M$) THEN GOSUB 8127
8120    IF A$<>"" THEN GOSUB 8138:GOTO 8124
8121    FOR Z=1 TO 100
8122    NEXT Z
8123 GOTO 8109
8124 LOCATE Y,X
8125 PRINT SPACE$(L+3);
8126 RETURN
8127 REM
8128 REM *** ERROR ***
8129 REM
8130 LOCATE Y,X+19
8131 PRINT "ERROR ..."
8132 FOR Z=1 TO 500
8133 NEXT Z
8134 LOCATE Y,X+19
8135 PRINT "         ":REM 9 ESPACIOS
8136 LET A$=""
8137 RETURN
8138 REM
8139 REM *** IMPRESION DE LA OPCION ELEGIDA ***
8140 REM
8141 LOCATE Y,X+19
```

Programa 2.

COMMODORE:

```
8017 POKE 214,Y:POKE 211,X
8109 POKE 214,Y:POKE 211,X+19
8110 PRINT CHR$(166);
8111 GET A$
8116 POKE 214,Y:POKE 211,X+19
8118 GET A$
8124 POKE 214,Y:POKE 211,X
8125 FOR Z=1 TO L+3:PRINT "   ";:NEXT Z
8130 POKE 214,Y:POKE 211,X
8134 POKE 214,Y:POKE 211,X
8141 POKE 214,Y:POKE 211,X
```

MSX:

```
8007 LOCATE X,Y
8109 LOCATE X+19,Y
8110 PRINT "_";
8116 LOCATE X+19,Y
8124 LOCATE X,Y
8130 LOCATE X+19,Y
8134 LOCATE X+19,Y
8141 LOCATE X+19,Y
```

SPECTRUM:

```
8107 PRINT AT Y,X;
8109 PRINT AT Y,X+19;
8110 PRINT CHR$(143);
8116 PRINT AT Y,X+19;
8124 PRINT AT Y,X;
```

```
8125 FOR Z=1 TO L+3:PRINT "  ";:NEXT Z
8130 PRINT AT Y,X+19;
8134 PRINT AT Y,X+19;
8141 PRINT AT Y,X+19;
```

Fig. 3. El programa 2 lo utlizaremos para recoger la opción elegida del menú.

El programa que viene a continuación es el último de la serie de rutinas que ya vimos en tomos anteriores. Esta rutina tiene el deber de imprimir el mensaje:

PULSA UNA TECLA

siempre que sea necesario hacer una pausa en el programa o llamar la atención del usuario.

```
8200 REM
8201 REM ********************
8202 REM * PULSA UNA TECLA *
8203 REM ********************
8204 REM
8205 LET A$="PULSA UNA TECLA"
8206 LET L=15
8208    BEEP
8209    FOR Z=1 TO 6
8210       LOCATE Y,X
8211       PRINT A$;
8212       IF INKEY$<>"" THEN GOTO 8222
8213       FOR K=1 TO 100
8214       NEXT K
8215       LOCATE Y,X
8216       PRINT SPACE$(L);
8217       IF INKEY$<>"" THEN GOTO 8222
8218       FOR K=1 TO 100
8219       NEXT K
8220    NEXT Z
8221 GOTO 8208
8222 LOCATE Y,X
8223 PRINT SPACE$(L);
8224 RETURN
```

Programa 3.

```
8300 REM
8301 REM ****************************
8302 REM * ORDENACION ALFANUMERICA *
8303 REM ****************************
8304 REM
```

Como siempre, las modificaciones que hay que realizar para que el programa pueda funcionar en ordenadores distintos al IBM y al AMSTRAD son las que se dan a continuación:

COMMODORE:

```
8208 REM
8210 POKE 214,Y:POKE 211,X
8212 GET T$:IF T$<>"  " THEN GOTO 8222
8215 POKE 214,Y:POKE 211,X
8216 FOR Z=1 TO L:PRINT "  ";:NEXT Z
8217 GET T$:IF T$<>  " " THEN GOTO 8222
8222 POKE 214,Y:POKE 211,X
8223 FOR Z=1 TO L:PRINT "  ";:NEXT Z
```

MSX:

```
8210 LOCATE X,Y
8215 LOCATE X,Y
8222 LOCATE X,Y
```

SPECTRUM:

```
8208 BEEP 0.2,30
8210 PRINT AT Y,X;
8215 PRINT AT Y,X;
8216 FOR Z=1 TO L:PRINT "  ";:NEXT Z
8222 PRINT AT Y,X;
8223 FOR Z=1 TO L:PRINT "  ";:NEXT Z
```

Fig. 4. Esta es una de las pantallas donde aparecerá el mensaje 'PULSA UNA TECLA'.

Una de las rutinas más importantes dentro de nuestro programa es la rutina de ordenación alfanumérica por campos. Esta rutina nos servirá para ordenar todo el fichero por el campo que nosotros deseemos. Antes de ver cómo funciona y para qué sirve, vamos a introducir el programa en el ordenador.

```
8305 CLS
8306 PRINT "O R D E N A R   F I C H E R O"
8307 PRINT "============================="
8308 PRINT
8309 PRINT "Ordenando fichero por el campo No. ";NC
8310 PRINT
8311 PRINT "ESPERA UN MOMENTO"
8312 LET CC=0
8313 FOR Z=1 TO TT
8314    IF MID$(F$(20),Z,1)=CHR$(254) THEN LET CC=CC+1
8315    IF CC=NC-1 THEN LET Z1=Z:LET Z=TT
8316 NEXT Z
8317 IF CC=NC-1 THEN GOTO 8328
8318 CLS
8319 PRINT "E R R O R"
8320 PRINT "---------"
8321 PRINT
8322 PRINT "Lo siento pero no he encontrado el campo No. ";NC
8323 PRINT
8324 PRINT "Estas seguro de que las fichas de este fichero tienen";NC;"campos?"
8325 X=1:Y=20:GOSUB 8200
8326 CLS
8327 RETURN
8328 FOR Z=1 TO NN
8329    FOR X=1 TO Z
8330       IF RIGHT$(F$(Z),Z1)<RIGHT$(F$(X),Z1) THEN LET A$=F$(Z):LET F$(Z)=F$(X
):LET F$(X)=A$
8331    NEXT X
8332 NEXT Z
8333 CLS
8334 PRINT "O P E R A C I O N   T E R M I N A D A"
8335 PRINT "====================================="
8336 PRINT
8337 PRINT "El fichero esta ordenado por el campo No. ";NC
8338 X=1:Y=20:GOSUB 8200
8339 CLS
8340 RETURN
```

Programa 4.

Ante todo, las modificaciones que hay que hacer para que el programa pueda funcionar en el SPECTRUM y en el COMMODORE son las siguientes:

COMMODORE:

8305 PRINT "<SHIFT-HOME>"
8318 PRINT "<SHIFT-HOME>"
8326 PRINT "<SHIFT-HOME>"
8333 PRINT "<SHIFT-HOME>"
8339 PRINT "<SHIFT-HOME>"

SPECTRUM:

8314 IF F$(20,Z)=CHR$(254) THEN LET CC=CC+1
8330 IF F$(Z,Z1 TO) <F$(X,Z1 TO) THEN LET A$=F$(Z)
:LET F$(Z)=F$(X):LET F$(X)=A$

Para que veas cómo funciona esta rutina te propongo que introduzcas el programa 5. Para que este programa funcione, tienes que unirlo con los programas números 3 y 4.

```
10 REM ********************************
20 REM * PROGRAMA DE DEMOSTRACION     *
30 REM * PARA VER EL FUNCIONAMIENTO   *
40 REM * DE LA RUTINA DE ORDENACION   *
50 REM ********************************
60 REM
```

```
 70 REM *** INSTRUCCIONES ***
 80 REM
 90 CLS
100 PRINT "INTRODUCE DIEZ NOMBRES DE PERSONAS CON SUS APELLIDOS"
110 PRINT
120 PRINT "TIENES QUE INTRODUCIRLOS DE LA SIGUIENTE MANERA:"
130 PRINT
140 PRINT "NOMBRE,PRIMER APELLIDO,SEGUNDO APELLIDO"
150 PRINT
160 PRINT "POR EJEMPLO:"
170 PRINT
180 PRINT "ANDRES,LOPEZ,RAMIREZ"
190 PRINT
200 PRINT "EMPIEZA A ESCRIBIR"
210 PRINT
220 REM
230 REM *** INTRODUCCION DE LOS NOMBRES ***
240 REM
250 DIM F$(10)
260 FOR I=1 TO 10
270    INPUT N$,M$,L$
280    LET F$(I)=N$+CHR$(254)+M$+CHR$(254)+L$
290    PRINT
300 NEXT I
310 REM
320 REM *** LLAMADA A LA SUBRUTINA ***
330 REM
340 CLS
350 PRINT "VAMOS A ORDENAR POR:"
360 PRINT
370 PRINT "1- NOMBRE"
380 PRINT "2- PRIMER APELLIDO"
390 PRINT "3- SEGUNDO APELLIDO"
400 PRINT
410 INPUT "INTRODUCE TU OPCION ";A$
420 IF A$="1" THEN LET NC=1
430 IF A$="2" THEN LET NC=2
440 IF A$="3" THEN LET NC=3
450 IF NC=0 THEN GOTO 400
460 GOSUB 8300
470 PRINT "AQUI ESTA TU FICHERO ORDENADO"
480 LET X$=CHR$(254)
480 FOR I=1 TO 10
490    LET N1=INSTR(F$(I),X$)-1
500    LET N2=INSTR(N1,F$(I),X$)-1
520    PRINT LEFT$(F$(I),N1);" ";MID$(F$(I),N1+2,N2);" ";RIGHT$(F$(I),N2
+2)
530 NEXT I
540 END
```

Programa 5.

Con el programa 5 podrás ver cómo una lista de nombres y apellidos es ordenada como tú desees. Aparecerá por la pantalla algo parecido a lo que hay en la figura 5.

Las modificaciones que hay que realizar para que funcione en el COMMODORE y en el SPECTRUM son las siguientes:

COMMODORE:

90 PRINT "<SHIFT-HOME>"
340 PRINT "<SHIFT-HOME>"

SPECTRUM:

250 DIM F$(10,50)
490 FOR J=1 TO LEN(F$(I))

Ordenado por nombre **Ordenado por el 1.er apellido** **Ordenado por el 2.º apellido**

Fig. 5.

```
491     IF F$(I,J)=CHR$(254) THEN LET
N1=J-1
492 NEXT J
500 FOR J=N1+2 TO LEN(F$(I))
501     IF F$(I,J)=CHR$(254) THEN LET
N2=J-1
502 NEXT J
520 PRINT F$(I, TO N1);" ";F$(I,N1+2 TO
N2);" "; F$(I,N2+2 TO)
```

El funcionamiento del programa 5 no es importante que lo conozcamos de momento. Más adelante, según vayamos avanzando en la realización del programa, veremos qué es lo que hace y cómo.

Del que sí vamos a hablar un poco es del programa 4. Este programa realiza la ordenación alfabética de todas las fichas del fichero por cualquiera de sus campos. Al principio del programa se mira si existe el campo por el que se ha de ordenar. En caso de no existir, imprime un mensaje de error.

Veamos su funcionamiento línea a línea:

Línea 8305. Borra la pantalla.

Líneas 8306 a 8311. Imprime en la pantalla un mensaje que nos informa que el ordenador está ordenando todo el fichero por un cierto campo. El número de campo por el que se va a ordenar viene dado por el valor de la variable numérica NC. Este valor será el número de campo que elija el usuario a la hora de ordenar.

Línea 8312. Inicializa la variable CC a cero. Esta variable se usará para saber si el número de campo por el que tenemos que ordenar es válido.

Línea 8313. Comienza un bucle, dentro del cual se va a comprobar que dicho campo existe.

Línea 8314. En esta línea preguntamos si el carácter al que apunta el puntero del bucle es un delimitador de campos. En caso afirmativo, se suma uno a la variable CC, que es la que se encarga de contar el número de campos del registro.

Línea 8315. Si el número de campos que se han contabilizado hasta el momento es igual al número de campo que tenemos que ordenar menos uno, entonces se acaba el bucle y se guarda en Z1 el valor actual del puntero.

Línea 8316. Termina el bucle de comprobación.

Línea 8317. Se pregunta si encontró el campo que estábamos buscando. En caso afirmativo, se sigue con la rutina.

Líneas 8318 a 8324. Si no se encontró el campo que buscábamos, se imprime un mensaje de error.

Línea 8325. Se imprime el mensaje PULSA UNA TECLA.

Línea 8326. Borramos pantalla.

Línea 8327. Y se devuelve el control al programa principal.

Línea 8328. Este grupo de líneas sólo se ejecuta si se encuentra el campo sobre el que tenemos que ordenar. En esta línea comienza un bucle dentro del cual se ordenará el fichero por el método de sustitución.

Línea 8229. Comienza otro bucle, que depende del primero, y cuya función es actuar como segundo puntero del fichero.

Línea 8230. Esta es la línea más importante de todo el programa. En ella se pregunta si el registro que apunta la variable Z es menor que el que apunta la variable X. En caso negativo, se continúa con el bucle, pero en caso afirmativo se intercambian ambos registros.

Línea 8231. Se termina el segundo bucle.

Línea 8232. Aquí se acaba el bucle principal.

Línea 8233. Borramos la pantalla.

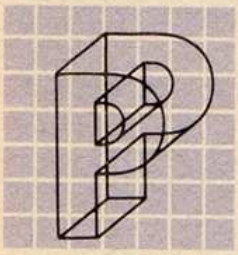
Líneas 8237. Se imprime una mensaje que informa al usuario del programa diciéndole que la operación está terminada.

Línea 8338. Se imprime el mensaje de PULSA UNA TECLA.

Línea 8339. Borramos la pantalla del ordenador.

Línea 8340. Y se devuelve el control al programa principal.

Circuitos de adaptación de señales



PARA la utilización de los circuitos de conversión analógico/digital y digital/analógico, descritos en el capítulo precedente, es conveniente presentar la forma idónea de adaptación de las señales a los niveles requeridos por los sistemas de conversión. Los circuitos que vamos a describir son diferentes formas de amplificadores, que utilizan como elemento principal un amplificador operacional. Para poder comprender las múltiples posibilidades de este tipo de circuito integrado empezaremos por una descripción básica de sus funciones y después describiremos circuitos específicos de aplicación para diferentes tipos de señales.

El amplificador operacional

Desde los principios de las aplicaciones electrónicas se conoce cómo conseguir amplificar señales de acuerdo con unos requerimientos dados por las necesidades específicas. La forma más común de conseguir una determinada curva de respuesta característica de un amplificador consiste en disponer de un amplificador lineal de ganancia elevada y realimentar la salida, a través de componentes pasivos adecuados, para que la respuesta en conjunto cumpla con las condiciones de diseño. Se reduce el problema de la realización del amplificador con característica específica a la del diseño de un circuito pasivo de realimentación que presente a la entrada del amplificador una parte de la señal amplificada de acuerdo con la curva característica a conseguir. El amplificador lineal que realiza la amplificación se denomina amplificador operacional y vamos a ver los rasgos que lo definen para que realmente sea la realimentación la que defina la característica.

La necesidad de diseñar equipos para muy diferentes aplicaciones con especificaciones estrictas de operación sobre señales ha conducido a la realización masiva de componentes que satisfagan en cualquiera de los casos los requerimientos de diseño. Así, puede decirse que la utilización de amplificadores operacionales integrados es la forma más eficiente y económica de resolver la mayoría de los problemas de tratamiento de señales analógicas.

El amplificador operacional ideal debe presentar las siguientes características:

— Ganancia de tensión infinita. Frecuentemente varía entre 20.000 y 10.000.000.

— Resistencia de entrada infinita. Pueden encontrarse circuitos con resistencias desde 100.000 ohmios hasta valores de muchos gigaohmios, utilizando transistores MOS en los circuitos de entrada.

— Resistencia de salida nula. Suele ser del orden de decenas de ohmios, pero por efecto de la realimentación puede hacerse despreciable.

— Ancho de banda infinito. Hay circuitos disponibles para aplicaciones de alta fre-

cuencia que pueden alcanzar cientos de megahertzios.

— Ausencia de limitaciones de la señal y de pérdidas.

Son varias las consecuencias inmediatas que se derivan de las propiedades ideales:

— La tensión necesaria a la entrada del amplificador, para cualquier valor de tensión de salida, es nula.

— La corriente a la entrada será nula, al ser infinita la resistencia de entrada.

A este comportamiento especial de presentar corriente nula con diferencia de tensión también nula se le denomina "masa virtual" y es de importancia fundamental para el funcionamiento y para el análisis de los circuitos diseñados con operacionales.

Los amplificadores operacionales en circuitos integrados actuales cumplen casi perfectamente las características ideales requeridas. Sin embargo, es necesario considerar algunas limitaciones que se presentan en la práctica dependientes de la tecnología empleada. Estas limitaciones son:

— Tensión de ajuste (offset) no nula y variable con la temperatura.

— Corriente de polarización (bias) no nula.

— Velocidad de subida (slew rate) limitada.

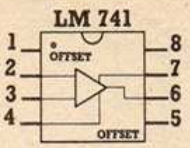
— Tensiones de saturación.

— Dependencia con las tensiones de alimentación.

— Ganancia en modo común no nula.

Para la mayoría de las aplicaciones es posible encontrar un circuito que satisfaga los requerimientos de ganancia y curva característica, dentro de un margen de temperatura limitado. Un amplificador operacional de gran difusión y que, por tanto, se encuentra a muy bajo precio es el 741. Puede encontrársele con diferentes prefijos, según el fabricante. Hay versiones que incluyen en el mismo circuito integrado hasta 4 amplificadores. Veamos sus características fundamentales y algunos ejemplos de aplicación para adaptación de señales a conversores A/D y D/A.

El 741 en su forma más común se presenta en circuito de 8 patillas. En la figura puede verse la denominación de cada una de ellas y las curvas principales descriptivas de su funcionamiento.



ESPECIFICACIONES

OFFSET: 2÷6 mV
BIAS: 80-500 nA
SLEW RATE: 0,5 V/μS
CMRR: 70-90 dB
GANANCIA: 20.000÷200.000
I_{SALIDA}: 20 mA

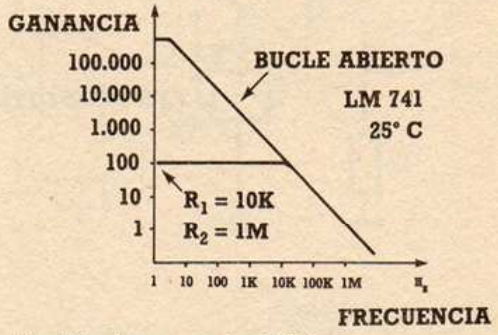


Fig. 1. Amplificador operacional 741.

La característica de amplificación en bucle abierto muestra una gran ganancia para valores de continua, que decrece gradualmente con la frecuencia. Este tipo de respuesta está diseñado expresamente para conseguir la estabilidad del amplificador, cualquiera que sea el circuito de realimentación empleada. El efecto real que produce es que es un poco lento para muchas aplicaciones, pero por lo demás, no es tan importante como que no oscile si el montaje no es todo lo cuidadoso que debiera. Por supuesto, que también pueden montarse osciladores con el 741, pero cuando realmente nos lo propongamos. La tensión de alimentación puede alcanzar los 30 voltios entre terminal positivo y negativo. Normalmente se emplea alimentación equilibrada, pero puede alimentarse con una sola fuente, teniendo entonces que desplazarse una de las entradas, resultando también desplazada la salida.

Existen numerosas variantes del circuito, manteniendo la compatibilidad entre patillas, con mejoras en algunas de las características: resistencia de entrada, corriente de salida, margen de temperaturas, ancho de banda, etc.

Amplificador inversor

El primer circuito que permite ver el funcionamiento del amplificador es el que se muestra en la figura 2.

La entrada de la señal se hace a través de R1. La realimentación se hace mediante R2, que se une a R1 en la entrada negativa. Un análisis del circuito ilustra el funcionamiento.

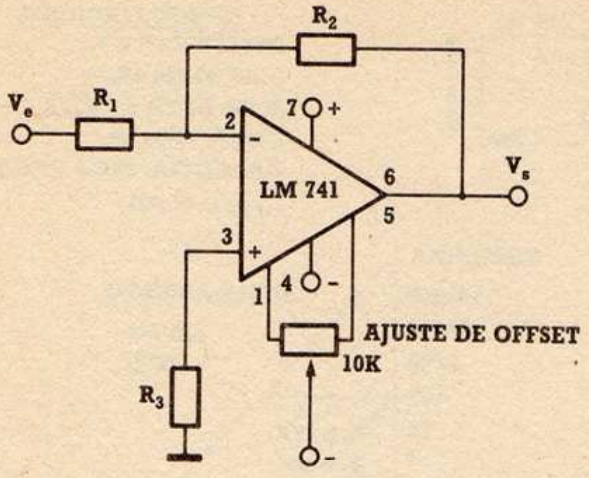


Fig. 2. Amplificador inversor.

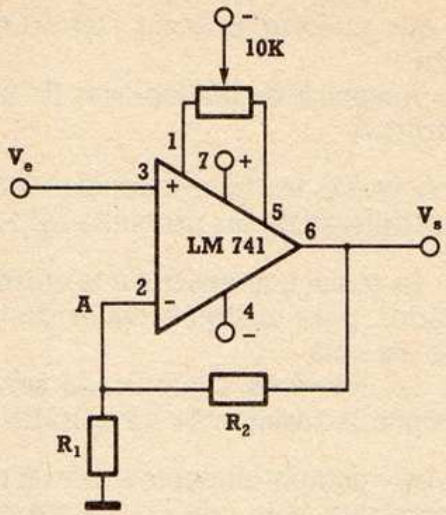


Fig. 3. Amplificador no inversor.

Para la tensión de entrada Ve, la corriente de entrada será Ve/R1, ya que la entrada del amplificador se comporta como masa virtual. La corriente en la realimentación ha de ser también la de la entrada, al no poder entrar nada en el amplificador. Por tanto, la tensión de salida deberá ser tal que las corrientes se igualen: -Vs/R2 = Ve/R1. La ganancia de tensión del circuito total será, pues: G = Vs/Ve = -R2/R1.

La importancia del resutado obtenido radica en que conseguimos la ganancia mediante cociente de los valores de dos elementos externos, que podremos seleccionar según las necesidades. El análisis simple empleado puede igualmente aplicarse al comportamiento en corriente alterna, si las resistencias de entrada y realimentación se sustituyen por impedancias en general o por otros circuitos cuya relación corriente/tensión pueda expresarse en estos términos.

La entrada + se suele conectar a masa a través de una resistencia R3 de valor igual al equivalente de R1 y R2 en paralelo, con objeto de eliminar los desplazamientos debidos al offset y las corrientes de bias.

Amplificador no inversor

Si conectamos la resistencia de entrada a masa y realizamos la entrada de la señal por la entrada no inversora, el comportamiento es ligeramente diferente. Para el amplificador operacional ideal la tensión en A deberá ser la misma que en la entrada positiva. Pero ese punto es el intermedio del divisor de tensión formado por R2 y R1. Luego VA = Vs R1/(R1+R2). La ganancia será entonces G = 1 + R2/R1. Un efecto lateral interesante de esta forma de montar el circuito es que la resistencia de entrada será la del amplificador operacional empleado, a diferencia del montaje en inversor, cuya resistencia de entrada viene dada por R1.

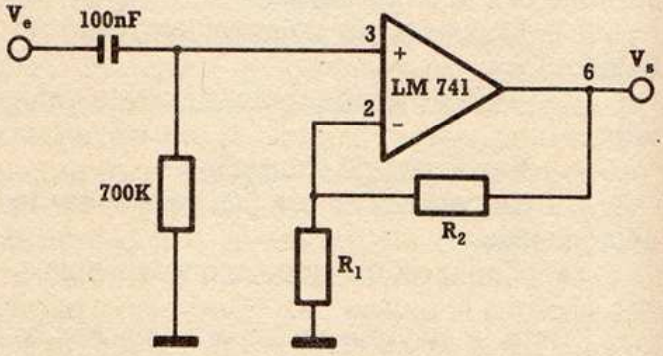


Fig. 4. Amplificador en alterna.

Amplificador en alterna

Estos mismos circuitos podemos emplearlos para amplificación de señales en corriente alterna, desacoplando la componente continua mediante condensadores, como se muestra en la figura. Una posible ventaja de utilizar estos montajes puede ser que al reducirse la ganancia en continua, el efecto de la tensión de offset prácticamente se elimina. La resistencia de 100 K que se coloca a la entrada es para acoplar impedancias con la etapa

precedente y para fijar la frecuencia de corte, junto con el condensador.

Amplificador diferencial

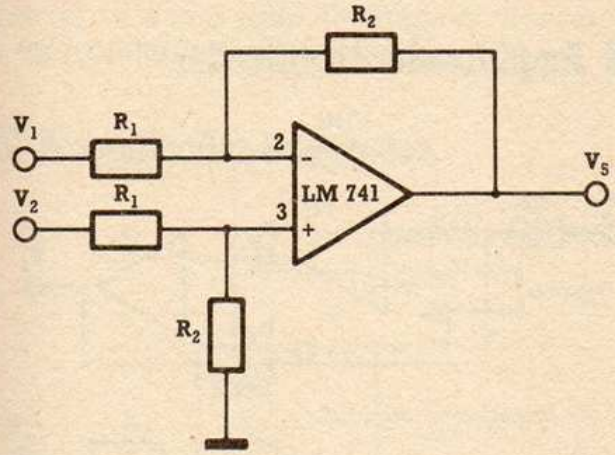


Fig. 5. Amplificador diferencial.

Para conseguir la diferencia amplificada de dos señales, montaremos el circuito de la figura. La tensión de salida es: Vs = (R2/R1)(V2-V1). Para que el circuito genere realmente la diferencia de las señales es necesario que las resistencias R1 y R2 de cada una de las ramas sean iguales. En la práctica convendrá comprar varias y seleccionar las más iguales o poner en una de las ramas una resistencia ajustable. El interés principal del montaje diferencial radica en que con frecuencia la señal a medir está superpuesta a otra señal más grande que afecta por igual a las dos entradas. Es necesario montar, entonces, un circuito que sólo amplifique la diferencia de las dos señales. La señal que afecta a las dos entradas se denomina el modo común. El amplificador operacional deberá ser insensible al modo común o por lo menos mucho menos sensible que al modo diferencial. Para cada amplificador operacional se suele indicar cuál es la ganancia en modo común o bien cuál es la relación de la ganancia en modo diferencial a la ganancia en modo común, denominándose entonces rechazo en modo común (CMRR).

Para aplicaciones que requieran un alto rechazo al modo común se monta el esquema de la figura, denominado amplificador de instrumentación. Presenta dos etapas de entrada en amplificador no inversor, seguidas de un amplificador en modo diferencial. Con este montaje se consigue independizar prácticamente la medida de la masa de referencia del

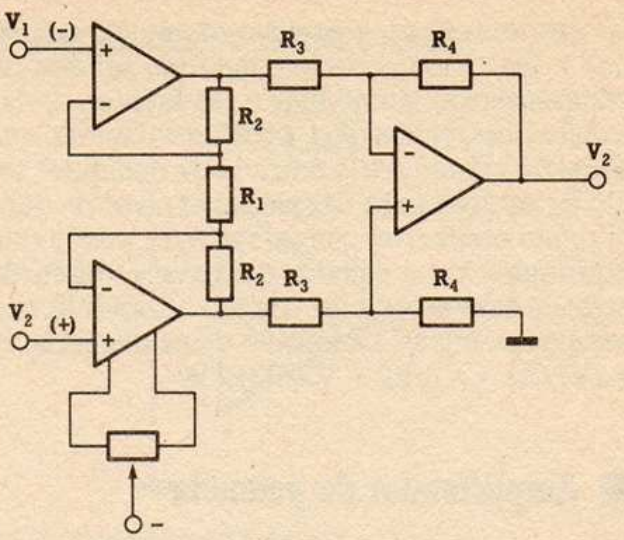


Fig. 6. Amplificador de instrumentación.

generador. Existen AO especiales para aplicaciones en las que es necesario disponer de un gran rechazo al modo común. Requieren un apareamiento muy exacto entre los componentes pasivos empleados. La ganancia para la configuración mostrada es: G = (1 + 2 R2/R1)(R4/R3), que puede ajustarse con un solo componente R1, si los demás están exactamente apareados. El ajuste de offset puede hacerse en uno de los AO de entrada o en ambos.

Amplificador sumador

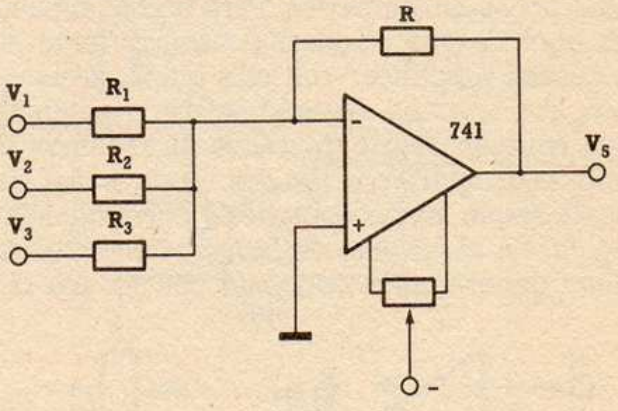


Fig. 7. Amplificador sumador.

Conociendo la propiedad de la masa virtual, podemos sumar corrientes a la entrada, con lo que la salida del amplificador será la suma de las tensiones de las entradas, con coeficientes inversamente proporcionales a los valores de las resistencias. Ya vimos este tipo de circuito al presentar el conversor digital/analógico y ahora vemos la explicación de su funcionamiento. Al ser independientes

las entradas entre sí, por ser el punto de unión una masa virtual, pueden añadirse señales de compensación sin afectar a las fuentes de las señales originales. Así podríamos sumar una señal de continua a la entrada de cualquier señal de alterna para desplazar el nivel de cero al punto medio del conversor A/D, con lo que podríamos tratar señales positivas y negativas con un conversor que trabaja solamente con señales positivas. La tensión de salida será: Vs = (V1/R1 + V2/R2 + V3/R3) / R.

Amplificador de potencia

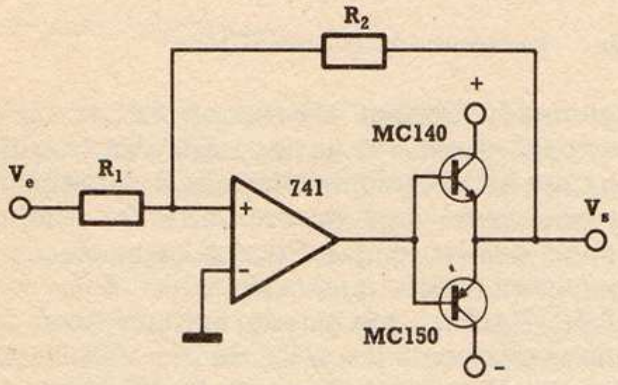


Fig. 8. Amplificador de potencia.

La corriente que pueden suministrar normalmente los amplificadores operacionales integrados es muy pequeña, del orden de unos pocos miliamperios. Para poder alimentar equipos que requieran más corriente es necesario amplificar con otra etapa. Si hacemos que la etapa que suministre la potencia esté también dentro del bucle de realimentación, conseguiremos las dos cosas: señal de salida según las necesidades y corriente suficiente. En el circuito de la figura se muestra cómo generar más corriente con un par de transistores complementarios de potencia. Para trabajar a baja frecuencia con el circuito mostrado es suficiente. Para alta frecuencia es necesario polarizar los transistores de salida, para eliminar la zona de transición.

Amplificador rectificador

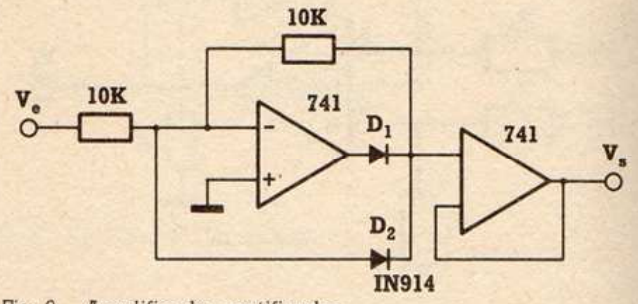


Fig. 9. Amplificador rectificador.

Para muchas aplicaciones es necesario tomar el valor absoluto de la señal de entrada o solamente la parte positiva o negativa. Mediante un diodo en el bucle de realimentación, conseguiremos que para valores positivos pase la señal por la rama del diodo D2 y para valores negativos intervenga el amplificador operacional invirtiendo la señal. Conviene poner un amplificador de ganancia unidad a la salida, pues realmente el rectificador no presenta baja impedancia de entrada cuando se alimenta con tensiones positivas.

Amplificador detector de pico

El valor que es necesario proporcionar al ordenador en muchas aplicaciones es el valor de pico de la señal durante un intervalo de tiempo. El circuito de la figura consta de un rectificador que carga un condensador. Des-

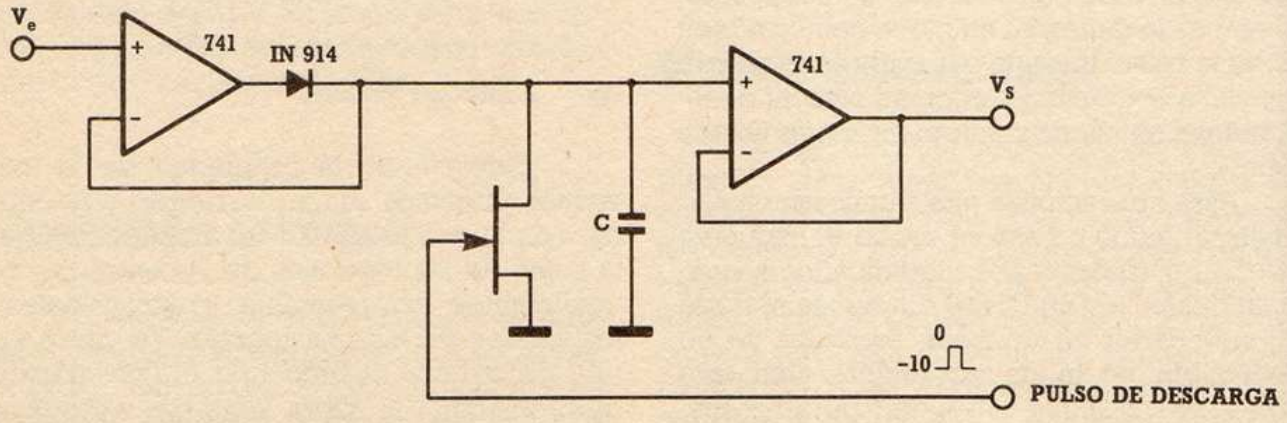


Fig. 10. Amplificador detector de pico.

pués de leído el valor, se descargará el condensador para prepararlo para otra medida. La descarga puede realizarse con un transistor en paralelo con el condensador, activándolo durante un tiempo suficiente. Conviene añadir detrás un amplificador seguidor, para aumentar todo lo posible el tiempo de descarga y así facilitar la conversión A/D.

Amplificador integrador

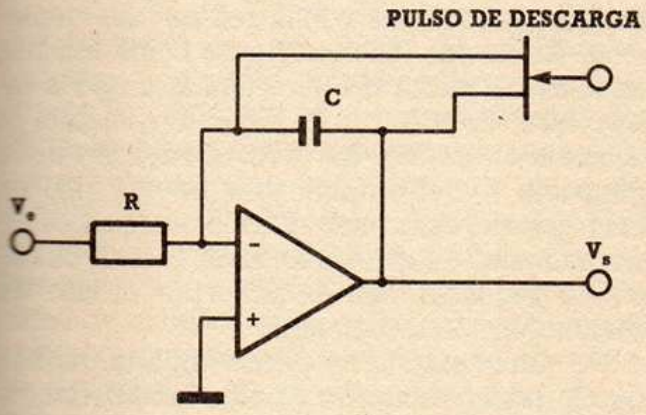


Fig. 11. Amplificador integrador.

Si en la rama de realimentación colocamos un condensador, el circuito generará una tensión creciente de salida, ante una tensión constante a la entrada, pues en el condensador la tensión es proporcional a la carga acumulada y ésta a su vez proporcional a la integral de la intensidad. En otras palabras, la tensión en el condensador es proporcional al producto de la corriente de entrada por el tiempo, que es una forma de considerar la realización de la integración matemática, si la variable independiente es el tiempo y la función a integrar es la señal de entrada. Esta consideración tan simple ha tenido resultados muy fructíferos en los primeros tiempos de la electrónica, pues mediante integradores, sumadores y unos pocos circuitos de control podía resolverse cualquier tipo de ecuaciones diferenciales lineales, sin más que plantear las ecuaciones de forma que se generaran las señales a partir de su derivada con un integrador, programando mediante potenciómetros los valores de los coeficientes. En la actualidad, con los ordenadores digitales, las soluciones se consiguen de forma mucho más precisa y sistemática, quedando los calculadores analógicos relegados a aplicaciones muy particulares relacionadas con sistemas en tiempo real.

Amplificador diferenciador

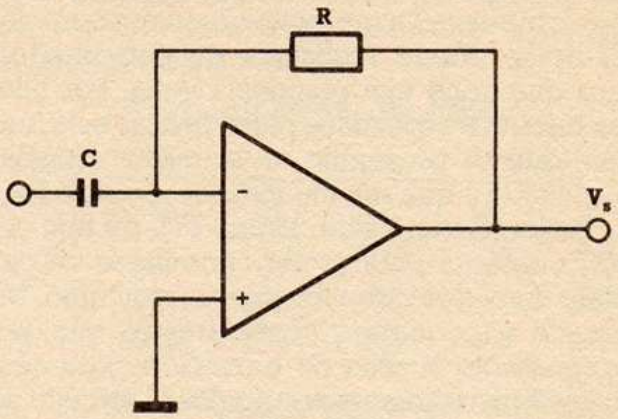


Fig. 12. Amplificador diferenciador.

Si la resistencia que sustituimos por un condensador es R1, el resultado es el complementario. El circuito realiza la derivada de la señal de entrada. En la práctica se utiliza mucho menos el circuito diferenciador, pues al ser amplificadas las altas frecuencias está sujeto a la influencia del ruido. Por ello, suele limitarse el ancho de banda incluyendo un pequeño condensador en paralelo con R y una resistencia en serie con C, de valores bajos.

Amplificador comparador

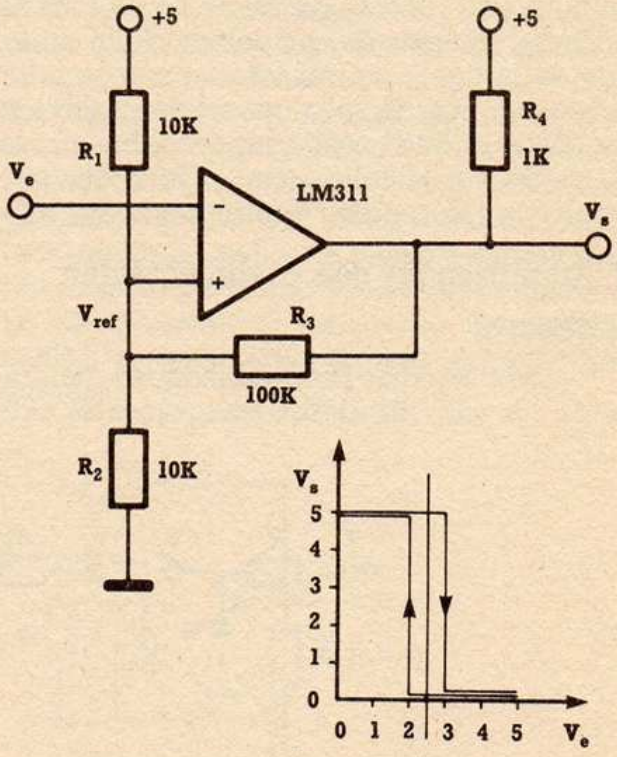


Fig. 13. Amplificador comparador.

Muchas veces es necesario determinar con precisión y rapidez cuál de dos señales es

mayor. El circuito de la figura da señal lógica 1 si la entrada Ve es menor que Vref, y 0 en caso contrario. Con el amplificador operacional típico puede realizarse un comparador, pero que tiene una respuesta lenta. Por ello, hay circuitos especiales para realizar esta función, que se presentan en el mismo formato que los AO y que son mucho más rápidos. Suelen proporcionar salida lógica TTL de tipo colector abierto, para poder conectarse en paralelo con otros circuitos lógicos similares. Un aspecto importante a considerar es que sea programable la zona de transición, para añadir histéresis si es necesario. En el circuito se indica cómo añadir histéresis, mediante la realimentación a la entrada positiva a través de R3 y la gráfica de su comportamiento.

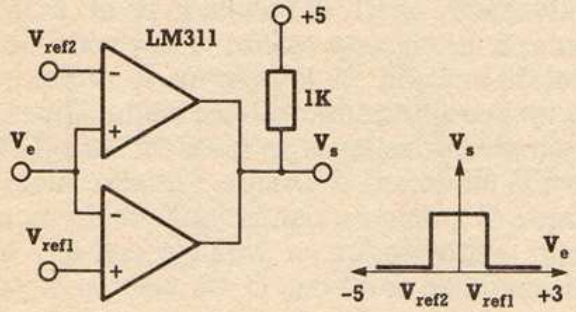


Fig. 14. Comparador de ventana.

Con ligeras modificaciones podemos hacer que la comparación tenga efecto para una determinada zona de valores. Esta zona se denomina "ventana de comparación". En el circuito de la figura se muestra un comparador de ventana que da señal de salida positiva si la señal de entrada está comprendida entre la referencia 1 y la referencia 2. Debe hacerse siempre la referencia 1 superior a la 2.

Amplificador con realimentación en general

Para estudiar las posibilidades generales de los amplificadores operacionales presentamos el circuito con realimentación en general. El circuito tiene dos partes: 1) el amplificador operacional, y 2) la red de realimentación. Aplicando las ecuaciones como hicimos en el amplificador inversor tenemos que la salida será: Vs = A (Ve–B Vs) y despejando el cociente G=Vs/Ve=A/(1+AB). Si el valor de la ganancia A del amplificador operacional es muy grande, como ocurre en la práctica, el resultado puede expresarse como: G=1/B, es decir, la ganancia está definida por el lazo de realimentación solamente.

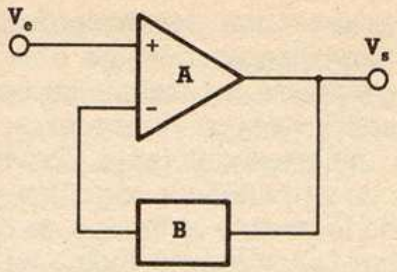


Fig. 15. Amplificador con realimentación en general.

Otros efectos no menos importantes son que la realimentación aumenta la impedancia de entrada y reduce la impedancia de salida, haciendo más ideal el comportamiento del circuito.

La aplicación fundamental de la realimentación con amplificadores operacionales es la realización de filtros activos. En conexión con ordenadores emplearemos filtros activos para acondicionar las señales a los niveles apropiados a nuestro sistema de conversión A/D. Por esta razón vamos a ver los circuitos fundamentales de filtros como ejemplos de aplicación.

Filtro de paso bajo

Un circuito sencillo que permite la atenuación de las altas frecuencias es el mostra-

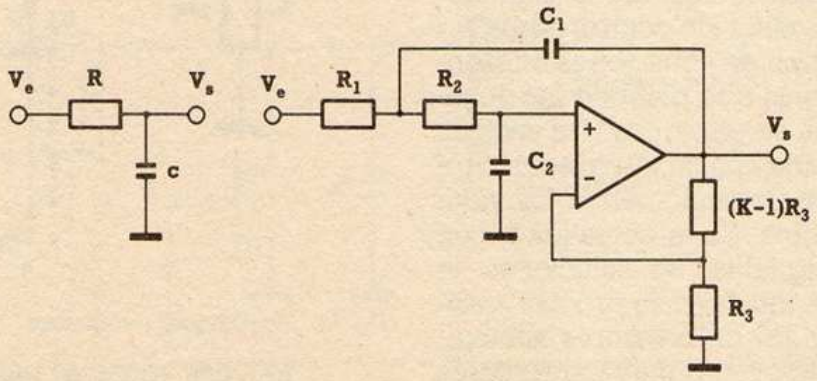


Fig. 16. Filtro de paso bajo.

do en la figura. Se comporta como un divisor de tensión en el que la rama de salida presenta impedancia que disminuye al aumentar la frecuencia. Por tanto, el resultado será que eliminará las frecuencias proporcionalmente a su valor, a partir de una frecuencia en la que se iguale la impedancia del condensador con la resistencia de entrada. Esta frecuencia viene definida por la fórmula: $F=1/2\pi RC$. Esta frecuencia suele denominarse frecuencia de corte y es aquélla para la que la amplitud de la señal se reduce a la mitad.

Mediante AO puede realizarse un filtro que presenta una atenuación de las frecuencias superiores más rápidamente que el filtro simple RC. Consiste en poner en cascada dos filtros iguales, pero el componente conectado a tierra del primero se conecta a la salida del amplificador. Si además se quiere amplificar la señal en la banda pasante, se monta la realimentación resistiva tomando una parte de la señal de salida. Con la estructura presentada puede conseguirse el comportamiento deseado del filtro según diferentes criterios de diseño:

— Transición de banda de paso a banda eliminada.
— Rizado en la banda pasante.
— Sobreoscilación de la respuesta a un impulso.

Estos criterios van ligados entre sí por las ecuaciones de diseño. El procedimiento normal consiste en seleccionar el tipo de respuesta deseado y calcular los valores de los componentes mediante consulta de las tablas apropiadas o la ayuda de programas de diseño asistido por ordenador. El número de etapas necesario viene dado por la pendiente en la zona de transición. Las tablas dan, de forma normalizada, los valores de los componentes y del factor de realimentación K.

En la tabla se indican los coeficientes para el diseño de filtros de los tipos de respuesta Butterworth(1), Bessel(2) y Chebychev(3) para 2 y 4 polos.

Tabla de diseño de filtros

Polos	B(1)		B(2)		Ch(3)
	K	fn	K	fn	K
2	1,58	1,27	1,27	1,23	1,84
4	1,15	1,43	1,08	0,60	1,58
	2,23	1,61	1,76	1,03	2,66

Por ejemplo, para realizar un filtro de paso bajo, hacemos C1=C2=C y R1=R2=R. Para la frecuencia de corte fc, definida por el valor de ganancia de tensión mitad, se calcula el producto $RC=1/2\pi fn$ fc. El valor de fn se deduce de la tabla para el tipo de respuesta seleccionada. El valor de R debe tomarse entre 5K y 200K, deduciéndose de la fórmula el valor de C. Del valor de K se calculan las resistencias de ganancia R3 y (K-1)R3.

Filtro de paso alto

El esquema básico para el filtro de paso alto es el complementario del de paso bajo, colocándose un condensador en la rama de entrada y una resistencia a masa. El efecto del circuito sobre la señal de entrada es el de favorecer las frecuencias altas, a partir de la frecuencia de corte definida por la expresión: $F=1/2\pi RC$. El montaje de filtro activo con amplificador operacional utiliza dos filtros en cascada de forma similar al montaje de paso bajo. Los cálculos para el diseño se hacen igualmente, considerando la frecuencia de corte normalizada con el valor inverso que en el caso anterior.

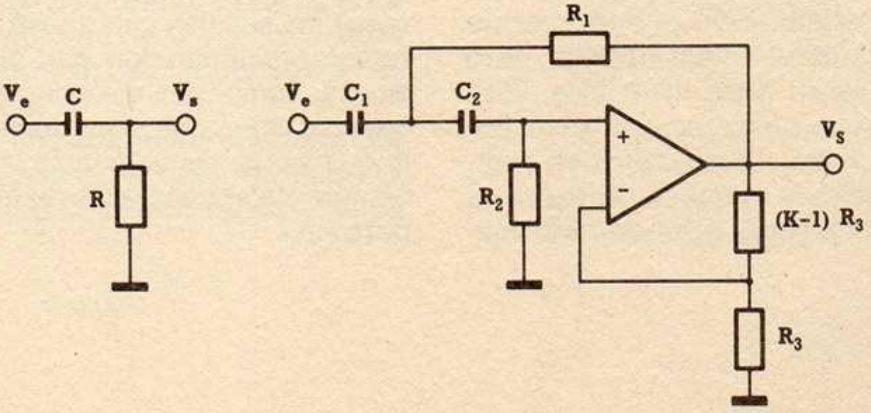


Fig. 17. Filtro de paso alto.

Filtro pasa banda

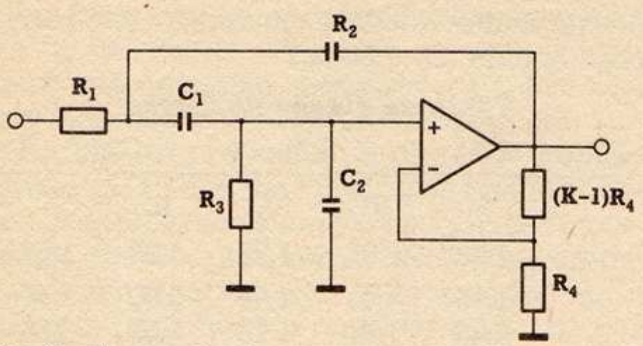


Fig. 18. Filtro pasa banda.

El circuito consta de dos partes, que realizan las funciones de filtro paso bajo y de paso alto, con frecuencias de corte que definen la gama de frecuencias que se transmite sin atenuación.

Se utiliza para seleccionar la señal de un sensor dentro del ruido que puede acompañarla, producido en el generador o en los cables de la transmisión.

Filtro eliminación de banda

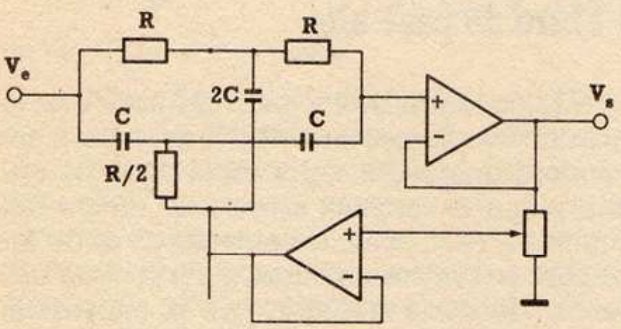


Fig. 19. Filtro eliminación de banda.

A veces es necesario eliminar un pequeño rango de frecuencias porque son producidas por algún elemento del circuito que es necesario emplear de forma inevitable. Así es el caso de la frecuencia de red, al utilizar sensores de muy bajo nivel de señal, que utilicen bobinas. La eliminación puede hacerse mediante filtro similar al de pasa banda, pero con las frecuencias de corte invertidas. Para frecuencias fijas suele utilizarse el filtro de doble T, que presenta una eliminación casi completa de la frecuencia a la que se sintoniza, si los componentes son de la precisión adecuada. Con el esquema de la figura puede igualmente controlarse la agudeza del filtrado, definida por el factor Q.

Recomendaciones de montaje

Para el correcto funcionamiento de los circuitos con amplificadores operacionales es conveniente tener presente las siguientes recomendaciones:

— Las alimentaciones deben ser estabilizadas, con un condensador próximo a los terminales de valor entre 100nF y 1microF.

— No debe sobrepasarse la corriente de salida admisible, aunque casi todos los OA actuales llevan protección a cortocircuito.

— Utilizar el ajuste de offset si el circuito debe trabajar en continua.

— No sobrepasar los límites de tensión diferencial admisible en las entradas, pues el circuito no lo "perdonaría" nunca.

— Si el circuito es de elevada ganancia, colocar siempre en paralelo con la resistencia de realimentación un pequeño condensador de 3-22 picofaradios, para evitar oscilaciones parásitas.

— Si se diseña con circuito impreso, reducir en lo posible el área de los terminales de entrada y su capacidad de acoplamiento con el terminal de salida, para reducir la posibilidad de oscilaciones.

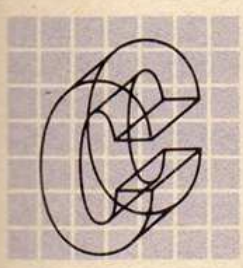
— Separar todo lo posible las zonas de AO de las de circuitos lógicos y usar diferentes alimentaciones y pistas de masa.

Conclusión

Se han presentado diversos casos prácticos de amplificadores de señal que nos permitirán adaptar cualquier tipo de generador al conversor A/D propuesto, para poder procesar las señales con nuestro ordenador. Veremos próximamente que alguna de las funciones, como, por ejemplo, el filtrado, puede hacerse igualmente mediante programa, con lo que podemos aumentar la flexibilidad y el número de aplicaciones de nuestro ordenador personal.

NATURALEZA Y TECNOLOGIA

Movimiento uniforme



CUANDO nos desplazamos en un viaje, por ejemplo, desde Madrid a Barcelona, jugamos al fútbol o simplemente corremos para alcanzar un autobús, estamos moviéndonos. En todos los casos pasamos de una posición a otra. Podría definirse, por tanto, el movimiento como el continuo cambio de posición o de lugar. Sin embargo, existe un segundo factor que influye en el movimiento, además del espacio o lugar. Si observamos el reloj al principio y al final de nuestro desplazamiento, veremos que el tiempo ha transcurrido. Para alcanzar el objetivo final del movimiento es necesario un cierto tiempo.

Existen otras magnitudes derivadas que influyen en el movimiento. No tardaremos lo mismo en un viaje de Madrid a Nueva York si viajamos en avión, en barco o si lo intentamos a nado. La velocidad es un elemento decisivo del movimiento.

Se trata de una magnitud derivada del espacio y del tiempo. Puede definirse como el espacio recorrido en la unidad de tiempo. Es decir:

$$v = \frac{s}{t}$$

donde:

v = velocidad s = espacio t = tiempo

El avión es más rápido que el barco, porque tarda menos tiempo en recorrer la misma distancia (Madrid-Nueva York).

En un movimiento hay que distinguir entre dos conceptos diferentes: **trayectoria** y **distancia.** Si vamos desde Lugo hasta Santander podemos elegir ir directamente o pasar por Madrid.

Fig. 1. Posibles trayectorias en un viaje de Lugo a Santander.

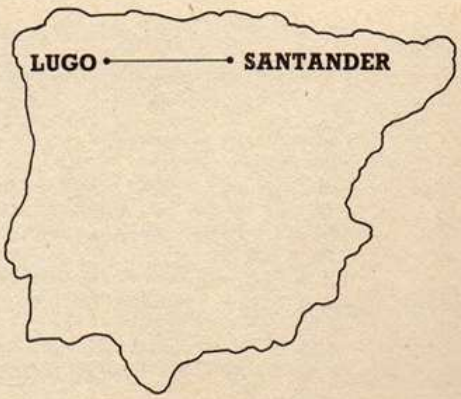


Fig. 2. Distancia en un viaje de Lugo a Santander.

En el primero de los casos la trayectoria será algo superior a 450 km., siendo mayor de 905 km. en el segundo. En todos los casos la distancia es de 450 km.

Existen diversos tipos de movimientos según sea la trayectoria. Podemos citar entre otros:

— Movimiento rectilíneo: la trayectoria es una recta.

— Movimiento circular: la trayectoria es una circunferencia.

— Movimiento ondulatorio: la trayectoria es una onda.

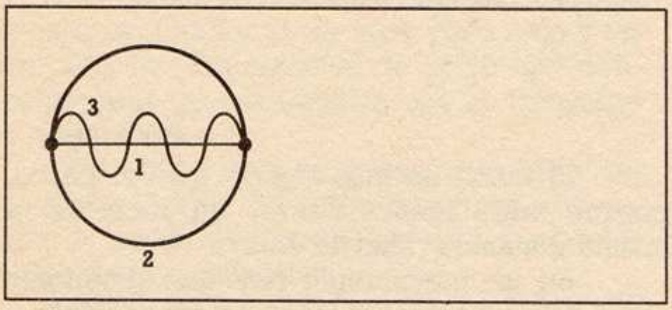


Fig. 3. Distintos tipos de movimiento:
1. Movimiento rectilíneo.
2. Movimiento circular.
3. Movimiento ondulatorio.

Podemos clasificar los movimientos según sea la velocidad en:

— Movimiento uniforme: la velocidad es constante.

— Movimiento acelerado: la velocidad es variable.

Si representamos gráficamente la variación del espacio en función del tiempo, veremos que en el caso del movimiento uniforme se obtiene una recta, mientras que en el movimiento acelerado obtendremos una curva.

El movimiento uniforme va a ser nuestro objeto de estudio de esta sección. Para ello hemos elaborado un programa en el que veremos desplazarse un coche a la velocidad que

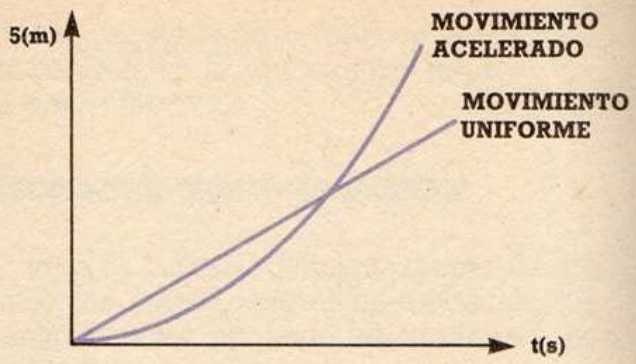


Fig. 4. Representación gráfica del movimiento uniforme y del movimiento acelerado.

elijamos. Al mismo tiempo podrá observarse un marcador de la distancia, velocidad y tiempo del movimiento. Además, se realiza una representación gráfica del espacio recorrido en función del tiempo. Cabe destacar que, lógicamente, cuanto mayor sea la velocidad elegida mayor será el espacio recorrido en la unidad de tiempo, y, por tanto, mayor será la pendiente de la recta que represente la variación del espacio en función del tiempo.

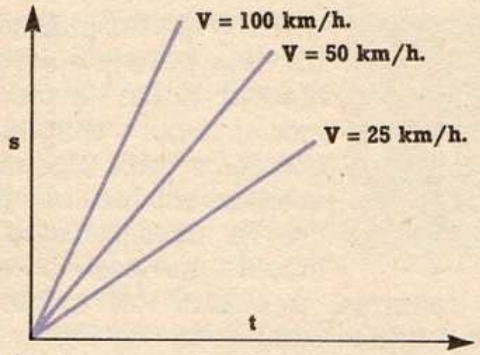


Fig. 5. Cuanto mayor sea la velocidad, mayor será la pendiente de la ruta.

Debemos señalar que el programa es muy diferente según sea el tipo de ordenador, dado que se manejan abundantemente gráficos. La versión primera es para AMSTRAD. En el caso de los ordenadores SPECTRUM e IBM se indican las variaciones. Las modificaciones para el MSX son muy semejantes a las del IBM.

```
10 REM ************************************
20 REM * PROGRAMA DE MOVIMIENTO UNIFORME *
30 REM *       VELOCIDAD CONSTANTE        *
40 REM ************************************
50 REM
60 REM ************************
70 REM * VALIDO PARA AMSTRAD *
80 REM ************************
90 MODE 2
100 REM ********************
110 REM * ENTRADA DE DATOS *
120 REM ********************
130 INPUT "velocidad en Km/h (10-100)";ve
140 IF ve<10 OR ve>100 THEN GOTO 130
150 CLS
```

```
160 REM ******************************
170 REM * DIBUJO DE EJES COORDENADOS *
180 REM ******************************
190 PLOT 100,150
200 DRAWR 300,0
210 PLOT 100,150
220 DRAWR 0,300
230 LOCATE 30,17:PRINT "t"
240 LOCATE 11,8:PRINT "s"
250 REM ***************
260 REM * MOVIMIENTO *
270 REM ***************
280 LET x=22:LET y=2
290 LET v=200-ve*2
300 GOSUB 1000
310 GOSUB 2000
320 GOSUB 1500
330 LET y=y+1
340 GOSUB 500
350 FOR z=1 TO v:NEXT z
360 IF y<75 THEN GOTO 300
370 END
500 REM ********************
510 REM * DIBUJO DEL COCHE *
520 REM ********************
530 LOCATE y,x-1:PRINT " ";CHR$(138);CHR$(131);CHR$(133);" "
540 LOCATE y,x:PRINT CHR$(143);CHR$(143);CHR$(143);CHR$(143);CHR$(143)
550 LOCATE y,x+1:PRINT " O O "
560 RETURN
1000 REM ***************************
1010 REM * TECNICA DEL MOVIMIENTO *
1020 REM ***************************
1030 LOCATE y,x:PRINT " "
1040 RETURN
1500 REM ************************************
1510 REM * CALCULO DE LA DISTANCIA Y TIEMPO *
1520 REM ************************************
1530 LET t=y/ve
1540 LET ti=INT(t*100)/100
1550 LOCATE 10,18:PRINT "velocidad:";ve;"Km/h"
1560 LOCATE 10,19:PRINT "distancia:";y;"Km"
1570 LOCATE 10,20:PRINT "tiempo:";ti:LOCATE 23,20:PRINT "h"
1580 RETURN
2000 REM ***********************
2010 REM * DIBUJO DEL GRAFICO *
2020 REM ***********************
2030 PLOT 100+t*75,150+y*3
2040 RETURN
```

Modificaciones para otros equipos

SPECTRUM

```
90 NO PONER
190 PLOT 100,100
200 DRAW 150,0
210 PLOT 100,100
220 DRAW 0,50
230 PRINT AT 10,20;"t"
240 PRINT AT 4,11;"s"
280 LET X=20:LET Y=1:LET T=0
360 IF Y<25 THEN GOTO 300
370 GOTO 9999
530 PRINT AT X-1,Y;"
";CHR$(133);CHR$(131);CHR$(138);" "
540 PRINT AT X,Y;CHR$(143);CHR$(143);
CHR$(143);CHR$(143);CHR$(143)
550 PRINT AT X+1,Y;" O O "
1030 PRINT AT X,Y;" "
1550 PRINT AT 16,5;"VELOCIDAD:";VE;"Km/h"
1560 PRINT AT 17,5;"DISTANCIA:";Y;"Km"
1570 PRINT AT 18,5;"TIEMPO:";TI;"h"
2030 PLOT 100+T*50,100+Y
```

IBM

```
90 SCREEN 1
190 NO PONER
200 LINE(50,100)-(320,100)
210 NO PONER
220 LINE (50,100)-(50,0)
230 LOCATE 14,26:PRINT "t"
240 LOCATE 6,5:PRINT "s"
280 X=20:Y=2
290 V=200-VE*2
360 IF Y<35 THEN GOTO 300
1030 LOCATE X,Y:PRINT " "
1550 LOCATE 16,10:PRINT
"VELOCIDAD:";VE;"Km/h"
1560 LOCATE 17,10:PRINT "DISTAN-
CIA:";Y;"Km"
1570 LOCATE 18,10:PRINT "TIEMPO:";TI;"h"
2030 PSET(50+T*100,100-Y*2)
```

Sobre el programa

En el programa pueden distinguirse tres partes:

— Diseño y movimiento del coche.
— Gráfico de la variación del espacio en función del tiempo.
— Visualización de distancia, tiempo y velocidad del movimiento.

1. *Diseño y movimiento del coche:*

Para el diseño del coche se ha empleado la técnica de agrupar varios caracteres existentes en el propio ordenador. En posteriores libros veremos cómo definir nuestros propios caracteres.

En el caso del AMSTRAD se han elegido los caracteres de código 131, 133, 138 y 143, además de dos oes para diseñar el conjunto entero.

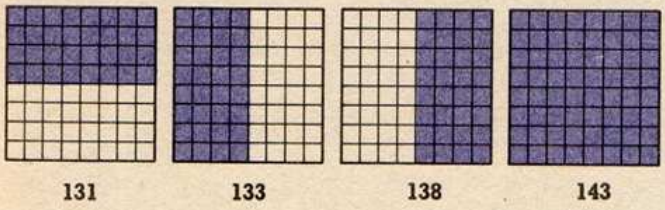


Fig. 6. Caracteres utilizados para el diseño del coche.

En total se emplean tres filas, cada una con 5 caracteres. Para diseñar el coche sólo es preciso "pegar" un carácter con el que le corresponde, de la misma forma que se resuelve un rompecabezas.

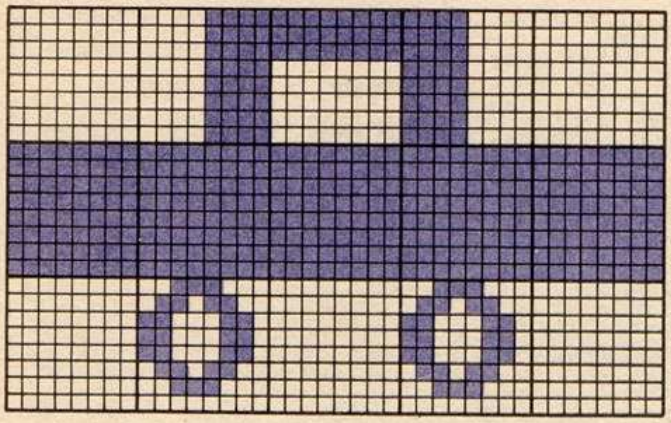


Fig. 7. Diseño del coche.

Para simular un movimiento en BASIC se utiliza una técnica parecida a la empleada para las películas de dibujos animados. Se trata de ir "dibujando" el coche a lo largo de todo el recorrido. Además, antes de dibujar el coche en la siguiente posición debemos borrar el coche anterior. Ello se hace visualizando espacios en blanco en el lugar que ocupaba el coche. En este programa, dado que el dibujo es regular y debido a que de las tres filas de caracteres, dos empiezan por espacios en blanco, sólo es necesario visualizar un blanco en la primera columna de la fila intermedia, con lo que el programa ganará en velocidad de ejecución.

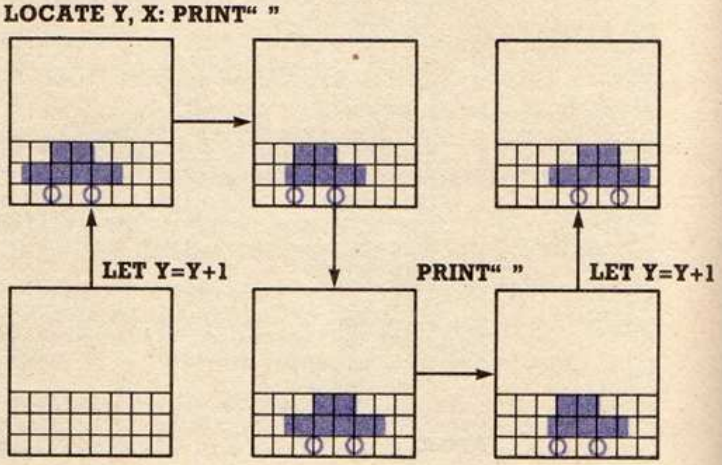


Fig. 8. Técnica del movimiento en BASIC.

2. *Gráfico de la variación del espacio en función del tiempo:*

Para visualizar la variación del espacio en función del tiempo se ha empleado un gráfico en alta resolución. Se utilizan las variables *t* para el tiempo e *y* para el espacio. Se representa *t* en el eje horizontal e *y* en el eje vertical.

3. *Visualización de la distancia, tiempo y velocidad:*

Simultáneamente al movimiento del coche y a la representación gráfica, se visualiza un contador de velocidad, distancia y tiempo. La velocidad es constante (movimiento uniforme) y se almacena en la variable VE. La distancia recorrida es la misma en todos los casos, aunque cuanto menor sea la velocidad se "recorrerá" más despacio. El tiempo se calcula en función de la distancia y la velocidad y se guarda en la variable **t.** Con el objeto de visualizar sólo una cifra con dos decimales se utiliza la variable **ti.** Este cálculo se realiza de manera estándar (en el AMSTRAD podría emplearse la función ROUND), multiplicando por 100, calculando la parte entera y dividiendo el resultado por 100.

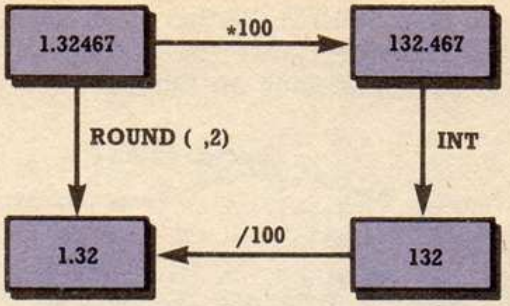


Fig. 9. Forma estándar de transformar un número en otro con dos decimales.

MATEMATICAS

Estadística: Cálculo de la media, varianza y desviación típica

Continuando con la estadística vamos a ver cómo podemos calcular diversos parámetros.

Cuando se realizan una serie de medidas, éstas pueden ser más o menos concordantes entre sí. Puede medirse esta "concordancia" mediante la varianza y la desviación típica.

La media aritmética no evalúa la diferencia entre los datos estadísticos. De todos es conocido que la renta "per cápita" es una medida de la media de ingresos. Sin embargo, en un país en el que existan grandes desequilibrios sociales, la media aritmética no será una buena forma de estimar la riqueza de un ciudadano. Este desequilibrio será fácilmente detectado por las medidas de dispersión, varianza y desviación típica.

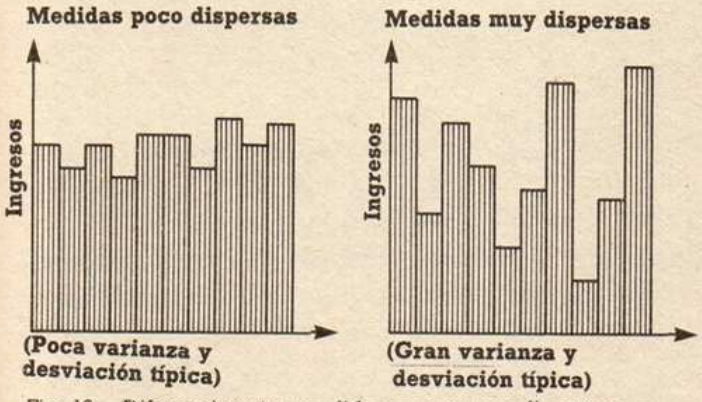


Fig. 10. Diferencia entre medidas poco y muy dispersas.

```
10 REM ************************************
20 REM * CALCULO DE MAGNITUDES ESTADISTICAS *
30 REM ************************************
40 REM
50 REM ************************************
60 REM *          VALIDO PARA             *
70 REM * AMSTRAD,SPECTRUM,IBM,MSX,COMMODORE *
80 REM ************************************
```

```
90 CLS:REM PONER PRINT CHR$(147) EN EL COMMODORE
100 REM ********************
110 REM * ENTRADA DE DATOS *
120 REM ********************
130 INPUT "NUMERO DE DATOS";N
140 CLS:REM PONER PRINT CHR$(147)EN EL COMMODORE
150 DIM D(N)
160 FOR I=1 TO N
170 PRINT "DATO:";I;
180 INPUT D(I)
190 CLS:REM PONER PRINT CHR$(147) EN EL COMMODORE
200 NEXT I
210 REM ************
220 REM * CALCULOS *
230 REM ************
240 LET S=0:LET S2=0
250 FOR I=1 TO N
260 LET S=S+D(I)
270 LET S2=S2+D(I)*D(I)
280 NEXT I
290 LET M=S/N
300 LET V=(S2-N*M*M)/(N-1)
310 LET DT=SQR(V)
320 REM ************************
330 REM * SALIDA DE RESULTADOS *
340 REM ************************
350 PRINT "DATOS"
360 PRINT "-----"
370 FOR I=1 TO N
380 PRINT I;"-";D(I)
390 NEXT I
400 PRINT:PRINT "NUMERO DE DATOS:";N
410 PRINT:PRINT "MEDIA:";INT(M*100)/100
420 PRINT:PRINT "VARIANZA:";INT(V*100)/100
430 PRINT:PRINT "DESVIACION TIPICA:";INT
    (DT*100)/100
```

El programa está hecho en un BASIC estándar. Lo hemos dividido en tres partes:

1. Entrada de datos.
2. Cálculo de las medidas estadísticas.
3. Salida de resultados.

Se ha utilizado una variable con subíndice denominada D(N) para guardar los datos. El cálculo de la media se hace utilizando un sumador (S), que se divide por el número de datos (N). El cálculo de la varianza se hace basándose en la siguiente fórmula:

$$V = \frac{1}{N-1}\left(\sum_{I=1}^{n} A(I)^2 - NM^2\right)$$

donde:

N = número de datos.
A(I) = datos de entrada.
M = media.

Para calcular la varianza empleamos un sumador de los cuadrados de los datos denominado S2. Posteriormente, y conocida la media, se utiliza la variable V para obtener el resultado final.

La desviación típica se determina extrayendo la raíz cuadrada de la varianza.

SOCIEDAD

Ríos del mundo: clasificación por longitudes

En esta sección vamos a construir una pequeña "base de datos" de los ríos mundiales, que clasificaremos en función de su longitud.

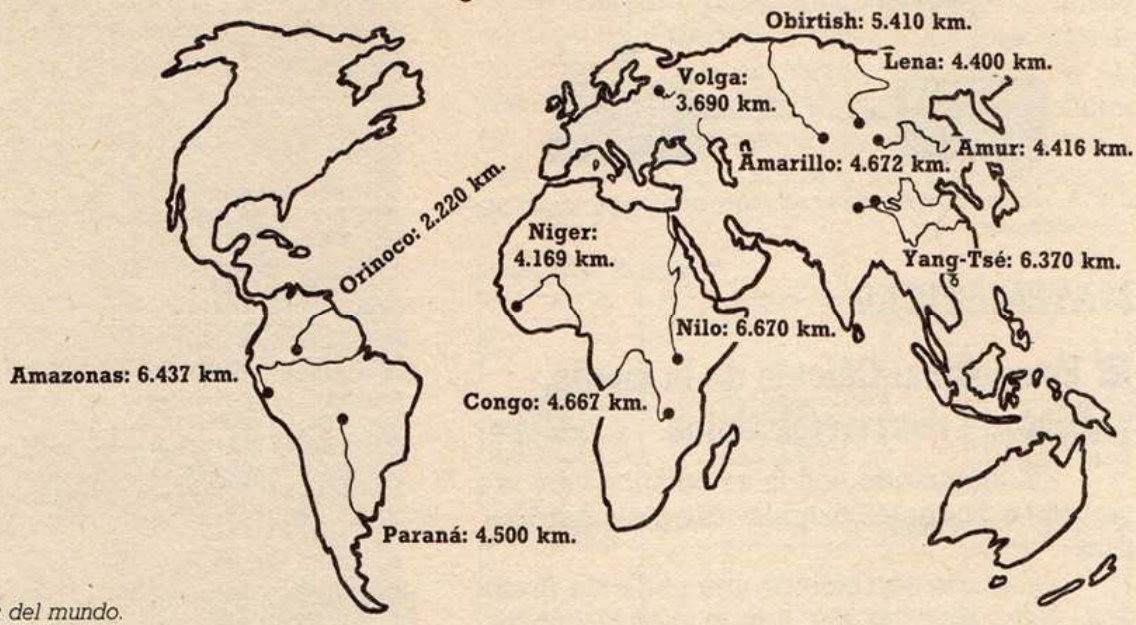


Fig. 11. Principales ríos del mundo.

```
10 REM ******************************
20 REM * PROGRAMA DE RIOS DEL MUNDO *
30 REM *   ORDENADOS POR LONGITUD   *
40 REM ******************************
50 REM
60 REM ******************************
70 REM * IBM,AMSTRAD,MSX,COMMODORE *
80 REM ******************************
90 CLS :REM PONER PRINT CHR$(143) EN COMMODORE
100 DIM R$(25,3):DIM C(25)
110 LET C=0
120 FOR I=1 TO 25
130 FOR J=1 TO 3
140 READ R$(I,J)
150 IF R$(I,J)="FIN" THEN GOTO 220
160 NEXT J
170 LET C=C+1
180 NEXT I
190 REM ************************
200 REM * PROCESO DE INDEXACION *
210 REM ************************
220 FOR I=1 TO C
230 FOR J=1 TO C
240 IF R$(I,3)<=R$(J,3) THEN LET C(I)=C(I)+1
250 NEXT J
260 NEXT I
270 REM ************************
280 REM * PROCESO DE BUSQUEDA *
290 REM ************************
300 FOR I=1 TO C
310 FOR J=1 TO C
320 IF C(J)=I THEN GOSUB 500
330 NEXT J
340 NEXT I
350 END
500 REM *****************
510 REM * VISUALIZACION *
520 REM *****************
530 PRINT I;"-";R$(J,1);TAB(15);R$(J,2);TAB(28);R$(J,3)
540 RETURN
1000 REM *********
1010 REM * DATOS *
1020 REM *********
1030 DATA "ORINOCO","AMERICA","2220"
1040 DATA "OB-IRTISH","ASIA","5410"
1050 DATA "AMAZONAS","AMERICA","6437"
```

```
1060 DATA "AMARILLO","ASIA","4672"
1070 DATA "NILO","AFRICA","6671"
1080 DATA "AMUR","ASIA","4416"
1090 DATA "VOLGA","EUROPA","3690"
1100 DATA "NIGER","AFRICA","4169"
1110 DATA "YANGTSE","ASIA","6370"
1120 DATA "CONGO","AFRICA","4667"
1130 DATA "PARANA","AMERICA","4500"
1140 DATA "LENA","ASIA","4400"
2000 DATA "FIN","FIN","0"
5000 REM *********************************
5010 REM * MODIFICACIONES PARA SPECTRUM *
5020 REM *********************************
5030 REM
5040 REM *********************************
5050 REM * 100 DIM R$(25,3,10):DIM C(25) *
5060 REM * 350 GOTO 9999                  *
5070 REM *********************************
```

Programa

El programa puede considerarse como una pequeña **base de datos** de los ríos del mundo.

Una base de datos puede definirse como un conjunto de datos relacionados entre sí y que están organizados en alguna forma. En nuestro caso están organizados en **campos** y **registros.** Se denomina registro al conjunto de datos referidos a un mismo elemento. Dentro de cada registro hay diversas informaciones. Cada uno de ellos será un campo.

Para guardar la base de datos se utiliza una matriz alfanumérica de dos subíndices denominada R$(I,J). El primer subíndice se refiere al número de registro. Llegará hasta el valor 12 (este número se calcula mediante el contador C). Se ha dimensionado hasta 25 para permitir la introducción de nuevos datos. Como último registro se ha empleado un "marcador", denominado "FIN",0,0, que permite saber cuándo se acaba la base de datos. El segundo subíndice se refiere al número de campo. En total tenemos 3 campos.

Para ordenar se emplea el método de **indexación.** Este método consiste en asignar un número de orden a cada registro, en función, en este programa, de la longitud del río en concreto.

Para indexar lo que hacemos es comparar cada registro con todos los demás en el caso del campo 3 (longitud del río). Si la longitud de un río es menor que la de otro, su número de orden será superior. Para guardar los números de orden utilizamos un contador con subíndice denominado C(I).

El método de indexación tiene la ventaja sobre una ordenación normal de que realmente no se ordena, sino que se deja la base de datos tal y como estaba desde un principio.

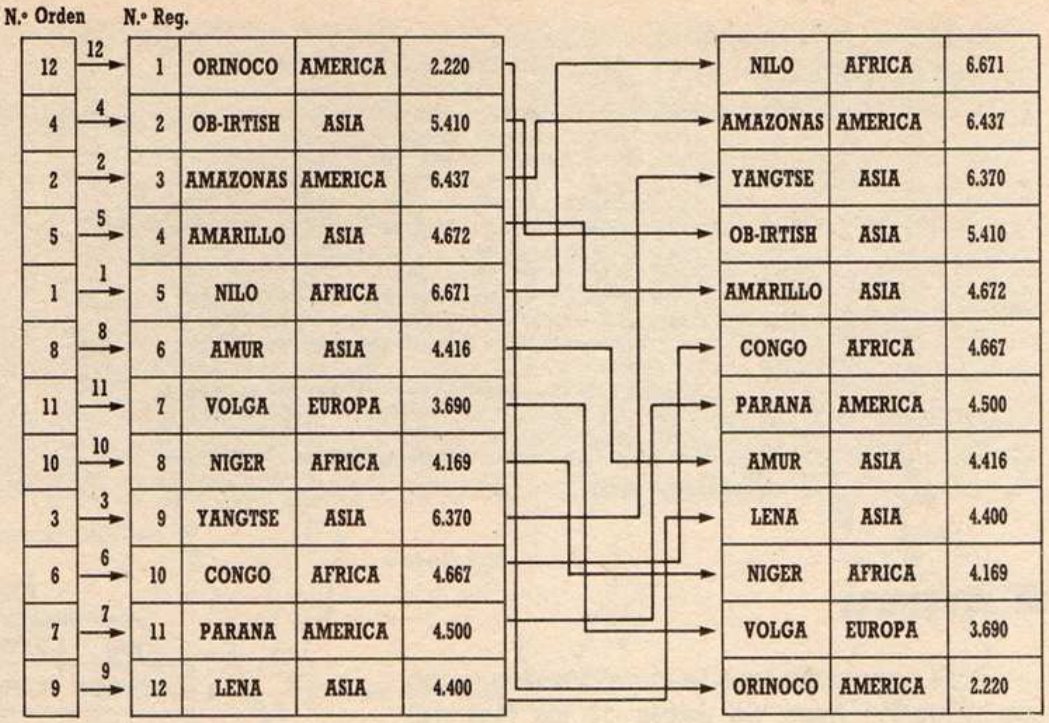
CAMPOS

REGISTROS	RIO	CONTINENTE	LONGITUD
1	ORINOCO	AMERICA	2.220
2	OB-IRTISH	ASIA	5.410
3	AMAZONAS	AMERICA	6.437
4	AMARILLO	ASIA	4.672
5	NILO	AFRICA	6.671
6	AMUR	ASIA	4.416
7	VOLGA	EUROPA	3.690
8	NIGER	AFRICA	4.169
9	YANGTSE	ASIA	6.370
10	CONGO	AFRICA	4.667
11	PARANA	AMERICA	4.500
12	LENA	ASIA	4.400

Fig. 12. Organización de una base de datos.

N.º Registro	N.º Orden
1	12
2	4
3	2
4	5
5	1
6	8
7	11
8	10
9	3
10	6
11	7
12	9

Fig. 13. Método de indexación.



N.º Orden	N.º Reg.				
12	12	1	ORINOCO	AMERICA	2.220
4	4	2	OB-IRTISH	ASIA	5.410
2	2	3	AMAZONAS	AMERICA	6.437
5	5	4	AMARILLO	ASIA	4.672
1	1	5	NILO	AFRICA	6.671
8	8	6	AMUR	ASIA	4.416
11	11	7	VOLGA	EUROPA	3.690
10	10	8	NIGER	AFRICA	4.169
3	3	9	YANGTSE	ASIA	6.370
6	6	10	CONGO	AFRICA	4.667
7	7	11	PARANA	AMERICA	4.500
9	9	12	LENA	ASIA	4.400

NILO	AFRICA	6.671
AMAZONAS	AMERICA	6.437
YANGTSE	ASIA	6.370
OB-IRTISH	ASIA	5.410
AMARILLO	ASIA	4.672
CONGO	AFRICA	4.667
PARANA	AMERICA	4.500
AMUR	ASIA	4.416
LENA	ASIA	4.400
NIGER	AFRICA	4.169
VOLGA	EUROPA	3.690
ORINOCO	AMERICA	2.220

Fig. 14. Proceso de búsqueda y visualización de la base de datos ordenada.

Si se desea visualizar la base de datos ordenada lo único que hay que hacer es ir buscando sucesivamente los números de orden.

En posteriores tomos veremos cómo ordenar por diferentes criterios. Además, realizaremos otras funciones de las bases de datos.

PARA LOS MAS JOVENES

Astrología: constelaciones del Zodíaco

En esta ocasión vamos a tratar el tema de las constelaciones del Zodiaco.

Las constelaciones son el resultado de un agrupamiento de estrellas que se asemejan a animales, objetos y personas. Esta semejanza es lo que da origen a sus nombres: Leo, Dragón, Escorpión. Entre todas ellas existen doce que forman una franja celeste, y que son recorridas por el Sol en su curso anual.

De ahí el que se asociara a la fecha de nacimiento una de las constelaciones del Zodiaco. Además, según sea el signo del Zodiaco de cada persona, así será su carácter, según los astrólogos.

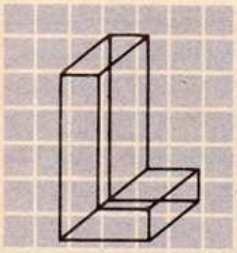
El siguiente programa sirve para establecer el signo del Zodiaco según sea nuestra fecha de nacimiento.

Para ello se introduce el mes y el día de nacimiento y se comparan sucesivamente con los datos del programa. En el momento en que se cumplan las condiciones sucesivas, se visualizará el signo correspondiente.

```
10 REM ************************************
20 REM * PROGRAMA DE SIGNOS DEL ZODIACO *
30 REM ************************************
40 REM
50 REM ****************************************
60 REM *              VALIDO PARA             *
70 REM * IBM,AMSTRAD,MSX,COMMODORE,SPECTRUM *
80 REM ****************************************
90 CLS :REM PONER PRINT CHR$(143) EN COMMODORE
100 DIM S$(12):REM PONER S$(12,12) EN SPECTRUM
110 DIM A(12):DIM X(12):DIM B(12):DIM Y(12)
120 FOR I=1 TO 12
130 READ S$(I),A(I),X(I),B(I),Y(I)
140 NEXT I
150 INPUT "EN QUE MES NACISTE";M
160 IF M<1 OR M>12 THEN GOTO 150
170 INPUT "EN QUE DIA";D
180 CLS :REM PONER PRINT CHR$(143) EN COMMODORE
190 FOR I=1 TO 12
200 IF M=X(I) OR M=Y(I) THEN GOSUB 500
210 NEXT I
220 END: REM PONER GOTO 9999 EN SPECTRUM
500 REM ******************************
510 REM * COMPROBACION DEL MES Y DIA *
520 REM ******************************
530 IF M=X(I) AND D>=A(I) THEN GOSUB 1000
540 IF M=Y(I) AND D<=B(I) THEN GOSUB 1000
550 RETURN
1000 REM ***************************
1010 REM * VISUALIZACION DEL SIGNO *
1020 REM ***************************
1030 PRINT "ERES DEL SIGNO:";S$(I)
1040 RETURN
2000 REM *********
2010 REM * DATOS *
2020 REM *********
2030 DATA "CAPRICORNIO",23,12,19,1
2040 DATA "ACUARIO",20,1,19,2
2050 DATA "PISCIS",20,2,20,3
2060 DATA "ARIES",21,3,19,4
2070 DATA "TAURO",20,4,20,5
2080 DATA "GEMINIS",21,5,21,6
2090 DATA "CANCER",21,5,21,6
2100 DATA "LEO",22,7,22,8
2110 DATA "VIRGO",23,8,22,9
2120 DATA "LIBRA",23,9,22,10
2130 DATA "ESCORPIO",23,10,21,11
2140 DATA "SAGITARIO",22,11,22,12
```

PEQUEÑA HISTORIA DE LA INFORMATICA

Las generaciones de ordenadores



OS ordenadores, desde su aparición, han sufrido un desarrollo incesante, pero discontinuo. En ciertos momentos de su historia han sufrido saltos importantes, para seguir mejorando linealmente sus posibilidades hasta otro "salto" posterior. Las causas de estos avances espectaculares en algunos puntos de su desarrollo se basan en la aparición de algunos componentes revolucionarios (tras un período más o menos largo de investigación y pruebas de laboratorio). Es verdad que en las primeras versiones innovadoras los resultados todavía tenían fallos susceptibles de mejora, pero lo importante es que esos "inventos" cambiaron la forma de enfrentarse con los problemas, y, en suma, revolucionaron las técnicas y componentes de los ordenadores electrónicos.

Atendiendo a estos "saltos" o cambios trascendentales en el caminar de los ordenadores, se pueden distinguir claramente cuatro "generaciones" de ordenadores, y una quinta, que ya está en marcha, y aunque todavía no haya llegado al empresario medio y al hombre de la calle, sí se encuentra en los laboratorios de desarrollo de las principales firmas de ordenadores (sobre todo japonesas).

Vamos ahora a repasar las cuatro generaciones de ordenadores. Sus diferencias, sus ventajas, etc.

Como ya sabe el lector, los primeros ordenadores eran enormes máquinas, llenas de cables y de otras máquinas accesorias (lectoras, verificadoras, etc.). Pero lo importante a retener en lo que se refiere a su tecnología es que eran máquinas que utilizaban válvulas. Las válvulas requieren para su funcionamiento tensiones de cierta magnitud, y de esto se deriva gran disipación de calor. Para remediar el problema se utilizaban sistemas de refrigeración de gran calidad y fiabilidad. Sin embargo, a pesar de los mimos y cuidados que recibían estas máquinas costosísimas, eso no impedía que las averías fueran bastante frecuentes (cada una o dos horas se producía

¿Sabía usted que...

Una cadena de montaje tiene mucha similitud con la organización de un ordenador.

a) Es muy útil organizar el trabajo total en pequeñas tareas sencillas (y a veces repetitivas).

b) La pieza (información) va pasando de un punto a otro de la cadena de montaje, sufriendo los distintos procesos.

c) (Miniaturización.) Cuanto más próximas estén las piezas del operario (o de la máquina que las va a montar) menor será el esfuerzo (energía) requerido, y además, la velocidad de montaje será mayor.

d) El resultado es una mejora considerable en la productividad.

Es lógico que sea de este modo, ya que se cumple una de las más viejas leyes de la naturaleza: la del máximo ahorro de energía.

una). Además, si consideramos el tiempo de proceso de una determinada operación, aunque el cálculo durara unas milésimas de segundo, como a este tiempo había que sumarle el de localización y reparación de la avería, el tiempo total resultaba muy largo.

En cuanto al coste no debemos olvidar que al precio altísimo de la máquina, también había que sumarle el del enorme gasto que suponían las averías (sustitución de piezas y salarios del nutrido y superespecializado equipo de mantenimiento). De todo lo expuesto se deduce que sólo organismos del Estado o empresas poderosísimas o paraestatales (universidades, etc.) podían permitirse la compra de un equipo de millones de dólares.

En lo que respecta a la arquitectura del ordenador, un dato importante es que los programas se ejecutaban de forma secuencial, es decir, en cada momento el ordenador sólo realizaba una tarea. Si, por ejemplo, estaba leyendo los datos, el resto de la máquina estaba parada, sin realizar ningún otro trabajo (proceso de la información, presentación de los resultados, etc.).

Resumiendo todo lo anterior, los ordenadores de la primera generación eran máquinas a válvulas, lentas, muy caras, muy delicadas y no muy fiables (averías).

De esta primera generación conocemos el Univac I, creado para la Oficina del Censo de los EE.UU. (la máquina se utilizó doce años). Este ordenador supuso el verdadero acicate para que la IBM entrara en el mercado de los ordenadores. El Univac I fue una máquina tremendamente popular. La popularidad no la ganó por su eficacia agilizando las tareas del censo, sino al ser mencionada en revistas, en programas de televisión y, sobre todo, al predecir el resultado de las elecciones americanas de 1952, mucho antes de contabilizar los votos.

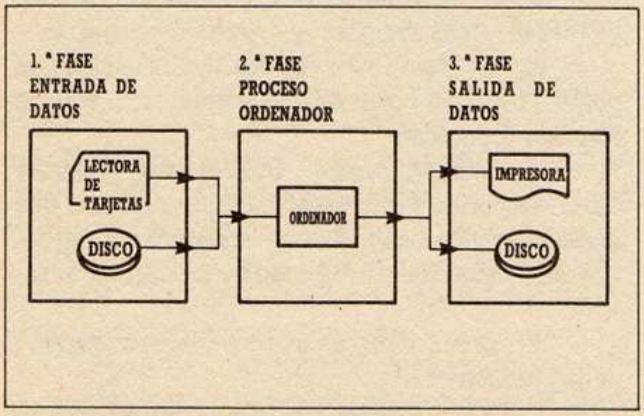
IBM no olvidó su enorme parque de máquinas de una sola tarea, tabuladoras, calculadoras, etc., pero, en el año 1953, sacó el ordenador 701. Su máquina no era tan innovadora como la Univac, y estaba bastante basada en otras máquinas de cálculo IBM que operaban con fichas. Sin embargo, esta gran empresa tenía una clientela enorme, a la que cuidaba y atendía con total dedicación, y eso le supuso la confianza de sus clientes para pasar de sus máquinas de una sola tarea a ordenadores. Probablemente, la decisión de fabricar ordenadores fue una decisión trascendental en la posición que actualmente ocupa la multinacional, ya que si hubiera retrasado esta decisión algún tiempo, la popularidad de Univac hubiera mermado sobremanera la aceptación posterior que tuvieron sus máquinas.

La segunda generación de ordenadores se caracterizó esencialmente por la sustitución de las válvulas por transistores. Este pequeño detalle hace que las máquinas de que hablamos (las de la segunda generación) sean sustancialmente distintas a las de la primera.

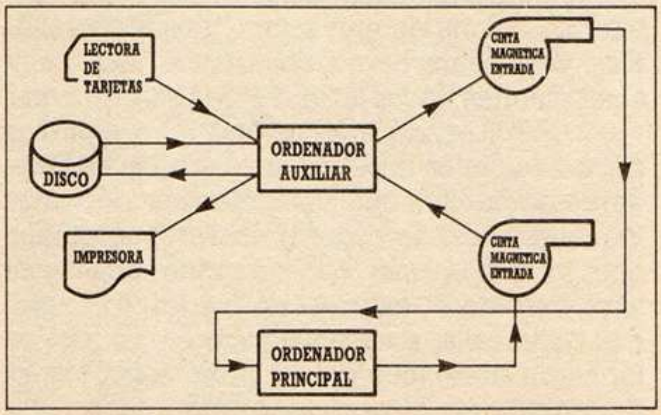
En primer lugar, los transistores trabajan a tensiones más bajas y además tienen una disipación de calor sustancialmente menor. Esta reducción hace que el papel de los sistemas de refrigeración no sea tan crucial, y que la tasa media de averías descienda considerablemente (de 10 a 1).

Los transistores son bastante menores en tamaño que las válvulas, y además iban montados sobre placas de circuito impreso de fácil fabricación (y que soportan sólo tensiones reducidas). La sustitución de las piezas era una tarea relativamente sencilla.

La reducción de tamaño también es un



Los ordenadores de la primera generación se caracterizan por ejecutar los programas de forma estrictamente secuencial, es decir, el ordenador sólo era capaz de realizar una sola tarea en cada momento.



La segunda generación de ordenadores surge cuando los transistores reemplazan a las válvulas. Estos nuevos ordenadores eran capaces de simultanear el cálculo con las operaciones de entrada y salida.

factor importante. De hecho, el gran impulso que experimentó la fabricación se debió en gran medida a las enormes ayudas que los Estados Unidos dedicaron a investigación de estas materias, como ayuda al Plan Espacial.

La fiabilidad también mejoró enormemente (un ordenador, en media, pasó a ser diez veces más fiable que los ordenadores de la primera generación).

Además, las plaquetas se podían cambiar fácilmente, sin necesidad de soldar y desoldar piezas, redundando en una fiabilidad mucho mayor, y un ahorro de tiempo espectacular.

Un factor importantísimo a considerar es el coste de la máquina. Este seguía siendo importante, pero ya era asequible a numerosas empresas comerciales. Al ser máquinas más fiables no requerían un equipo de mantenimiento tan completo como los ordenadores de la anterior generación. El coste es un factor muy importante en el desarrollo de los ordenadores, porque sin un mercado serio, la investigación se habría reducido muchísimo, y los avances posteriores habrían sido mucho más lentos.

Otra característica que mejoró enormemente fue la velocidad. Como antes hemos indicado, el tiempo perdido en reparaciones se había reducido a la décima parte. Además, la arquitectura del ordenador era tal, que permitía que dentro de un mismo programa los cálculos se pudieran simultanear con las operaciones de entrada y salida, lo que suponía un aumento considerable en la velocidad. Esta característica a primera vista puede parecer un avance enorme y, sin embargo, no era todo lo útil que cabía esperar, ya que las unidades de entrada y salida seguían siendo bastante lentas, y durante la presentación de resultados de un programa no se podían realizar, por ejemplo, cálculos relativos a otro programa.

Más adelante se mejoraron considerablemente los sistemas de almacenamiento permanente, apareciendo las cintas magnéticas. Esto supuso la utilización de otro ordenador auxiliar mucho más pequeño, que trabajaba en colaboración con el ordenador principal. Las tareas a realizar se agrupaban en "lotes" que iban ejecutándose siguiendo un orden de prioridad establecido. Estos lotes se organizaban en forma de "cola".

Naturalmente, esta "cola" era controlada para que siguiera la prioridad establecida. Más adelante también aparecerá otra "cola" de presentación de resultados o "spool".

La velocidad aumentó bastante con este sistema, pero el aprovechamiento de la máquina tampoco era todo lo bueno que sería de desear, ya que para conocer y utilizar los resultados era necesario esperar a que el ordenador central terminara todos los procesos del lote.

Algún tiempo después, la operación por lotes podrá realizarse bien con un único ordenador, marcando prioridades, o bien con dos, uno principal que efectúa los cálculos, ejecuta el programa de ese momento, y otro secundario, que es el que controla la "cola". Los resultados también pueden ser enviados al sistema de almacenamiento de cinta magnética.

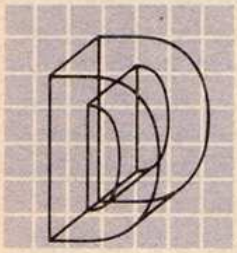
Avanzando aún más en el tiempo, la misma operación se efectuarán accediendo al ordenador principal desde diferentes terminales y siguiendo también un sistema de prioridades, con colas de entrada y de salida ("spool").

Al abrirse el mercado de ordenadores a empresas comerciales, comenzaron a surgir modelos de distintos precios y características, de forma que el usuario podía seleccionar el ordenador que se ajustara más a sus necesidades. Así, IBM sacó la serie 7000, de la que el más popular fue el 7090. Las otras marcas importantes también ofrecían modelos como el Univac III, Honeywell 800, CDC 3600 de Control Data, etc.

En lo que respecta a la programación, los lenguajes no estaban muy desarrollados, eran lenguajes de bajo nivel, sólo asequibles a programadores muy especializados. Sin embargo, las grandes firmas ofrecían las máquinas con un software que se podía adaptar al de otros equipos, si en otro momento el usuario se decidía a comprar otro ordenador más potente de la misma marca. En muchos casos los programadores que realizaban los cambios eran personal de la empresa fabricante, facturando ésta más tarde al usuario el tiempo utilizado en las transformaciones.

Este trato tan personalizado entre empresa fabricante y cliente se debía principalmente a que no existía compatibilidad alguna entre las distintas empresas fabricantes. Así, pues, era muy importante ampliar la clientela, ya que una vez obtenido el cliente, como los lenguajes de programación eran de bajo nivel (muy próximos al lenguaje máquina), resultaba mucho más sencillo comprar una máquina más potente a la misma marca, que otra de las mismas características de una firma diferente. Las conversiones eran mucho más sencillas.

Robótica II



E acuerdo con los sistemas de coordenadas indicados, y con los tipos de articulaciones disponibles, las posibilidades de organización del manipulador de los robots son enormes, pero, de hecho, no todos los tipos son igualmente utilizados; la figura 5 muestra cómo la mayoría de los RI instalados en el mundo operan en un sistema de coordenadas cilíndricas (el 58,5 %). Además, los cuatro sistemas básicos (rectangular en el plano, y en el espacio, cilíndricas y esféricas) suponen un 90 % del total de los sistemas en funcionamiento.

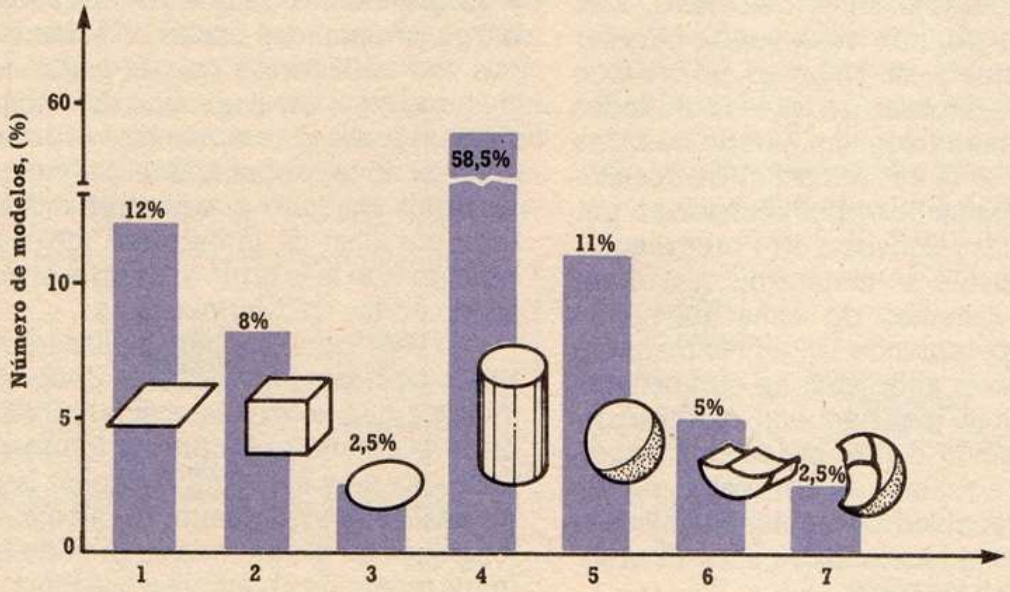


Fig. 5. Distribución de los modelos de robot según el sistema de coordenadas en el que realiza los movimientos básicos el sistema mecánico (manipulador). 1, rectangulares planas; 2, rectangulares en el espacio; 3, polares; 4, cilíndricas; 5, esféricas; 6, angulares cilíndricas; 7, angulares esféricas.

En cuanto al número de grados de libertad del sistema (que indica la flexibilidad de la estructura que se utiliza), hay que decir que la mayoría de los robots fabricados en el

mundo disponen de 4 ó 5 (un 63 % del total), y que prácticamente ninguno (un 1,5 %) utiliza siete o más grados de libertad. En la figura 6 puede verse la distribución de robots en función del número de grados de libertad que poseen.

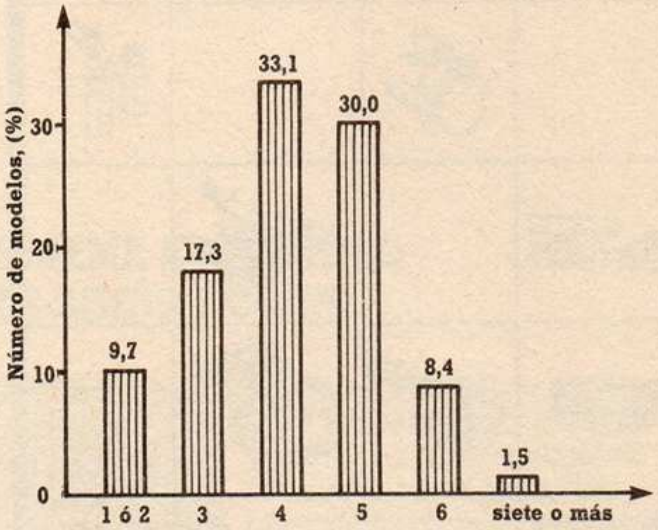


Fig. 6. Distribución de los modelos de robots existentes en el mundo, según los grados de libertad de que disponen.

Referente a los elementos terminales, la variedad es enorme. Existen unos tipos clásicos de terminales (e incluso intercambiables, estandarizados), pero en numerosas ocasiones se diseñan de un modo específico para una determinada tarea o actividad.

En estos elementos terminales, pueden existir diferencias en el sistema de agarre y sujeción de las piezas, teniendo:

a) Mecanismos puros de agarre, que sujetan los objetos mediante una acción cinemática de los elementos de agarre (patas, dedos articulados, mordazas...). Estos mecanismos producen un bloqueo combinado con una fuerza de fricción (suelen distinguirse incluso entre los sistemas meramente mecánicos y aquéllos que disponen de cámaras elásticas expandibles mediante la inyección en su interior de aire o agua).

b) Mecanismos de soporte (sujeción mediante garfios, ganchos de izamiento, cucharas contenedoras) que sujetan los objetos por algún asa o asidero, o bien que lo elevan en su interior mediante espátulas o soportes.

c) Mecanismos de arrastre de piezas (tirando de ellas) mediante una fuerza de tracción producida por diversos fenómenos físicos (los más usuales son: cámaras o recipientes de vacío y dispositivos de atracción electromagnética).

Por otro lado, la tarea a realizar implica la necesidad de utilizar un tipo de elemento terminal u otro; existen elementos capaces de:

a) Relocalizar objetos cambiando su posición (suelen ser verdaderos brazos, parecidos a los humanos, con dedos controlables).

b) Alinear (y centrar) los objetos de acuerdo con ejes de simetría o líneas de referencia.

c) Posicionar las piezas en superficies predeterminadas.

d) Reubicar los objetos (es decir, éstos pueden retener la posición en que se encontraba un elemento antes de su manipulación para reubicarlo posteriormente), etc.

En la figura 7 se muestran algunos de los elementos terminales utilizados usualmente en los robots, clasificados según el mecanismo de sujeción.

Por último, respecto de la programación de un robot hemos de decir que existen dos procedimientos generales de programación (o enseñanza) de un robot: la llamada programación gestual y la programación mediante un verdadero lenguaje de programación.

1. En la programación gestual se enseña al robot directamente a hacer los movimientos que luego ha de realizar.

El robot dispone de elementos suficientes para retener ("memorizar") los movimientos realizados, y poder, posteriormente, repetir la secuencia.

La presentación al robot de los movimientos básicos puede hacerse de dos modos: bien moviendo físicamente al robot a las posiciones adecudas de un modo manual, bien mediante un dispositivo de órdenes (accionador, consola de control, etc.) que hace que el robot se mueva y una vez establecida la secuencia de acciones a realizar, se memorice. Esta segunda opción se utiliza cuando no es fácilmente accesible la mano del robot (porque está alejada, o alta..., o porque está en un ambiente hostil, radiactividad, contaminación, etcétera).

En cuanto a la opción de movimiento físico del robot, hay que indicar que, en ocasiones, se trataba con un «maniquí», más fácilmente manipulable que el propio robot.

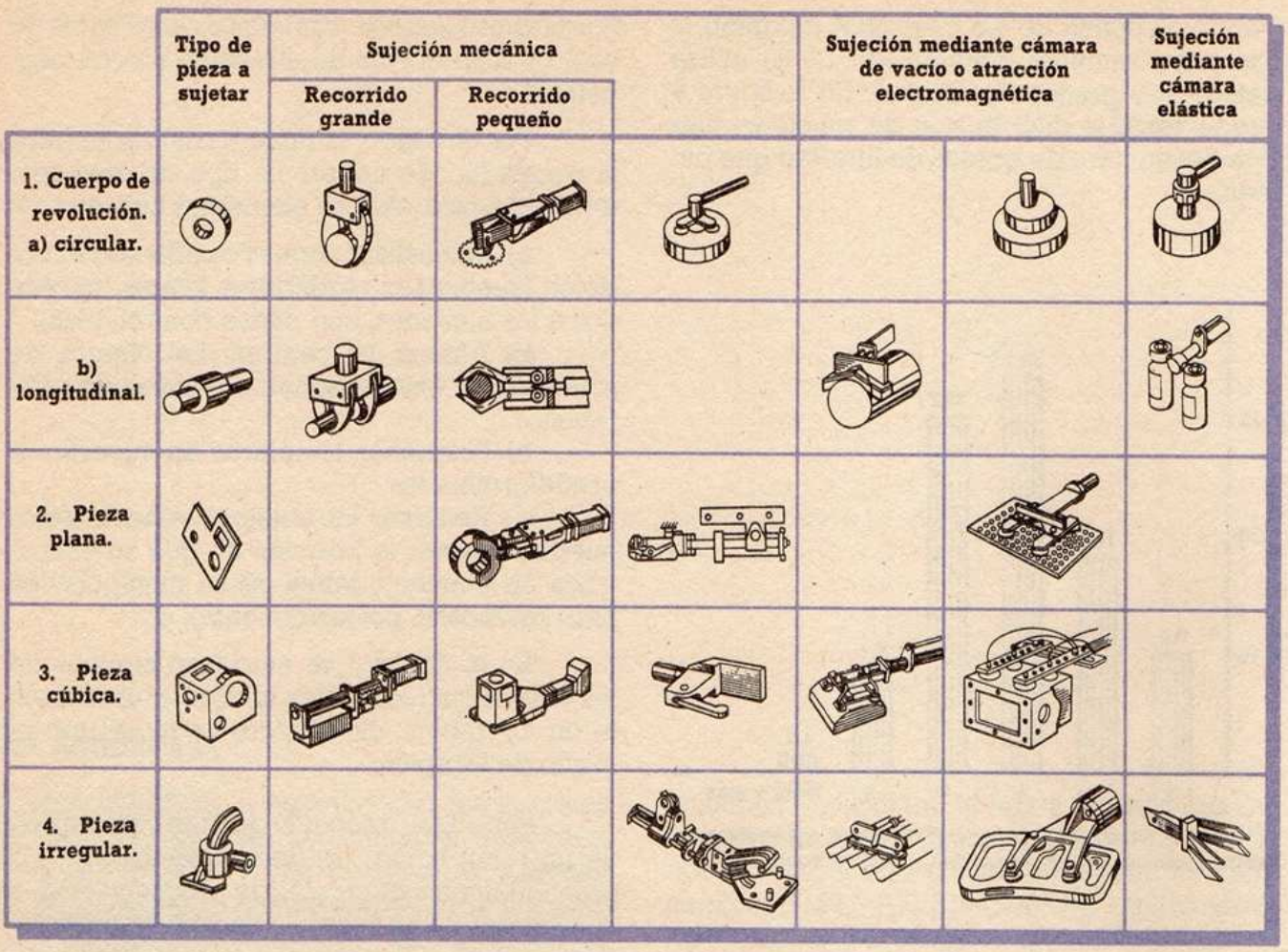


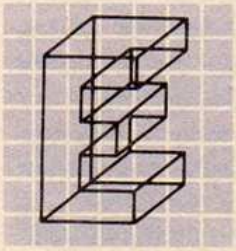
Fig. 7. Diseño de elementos terminales de los manipuladores de los robots.

c) La programación propiamente dicha se realiza utilizando lenguajes específicos (aunque vinculados a los lenguajes generales de programación, en los que están escritos) que incorporan las funciones gráficas y de desplazamiento del sistema articulado del robot. La programación es complicada, porque es difícil la determinación de los puntos y de las posiciones (en el sistema de coordenadas que utilice el manipulador del robot) del elemento terminal.

Es usual utilizar para la "enseñanza" del robot un sistema mixto en que las posiciones y acciones básicas se le muestran por un proceso de tipo "gestual", y la secuencia general y las condiciones de control se le indican mediante órdenes.

EJEMPLO PRACTICO DE ROBOT INDUSTRIAL

SISTEMA AUTOMATICO DE DISEÑO Y CORTE



L sistema automático de diseño de patrones, marcada y corte, ha sido concebido como una ayuda importante para multitud de industrias, principalmente de confección, aunque también es útil para las empresas de calzado, de tapicería, velas, etc.

Este tipo de sistema consta de los siguientes elementos:

— Un ordenador PC-AT, con una memoria de 1,5 megabytes.

— Un monitor en color con una pantalla de alta resolución de (1024 × 1024) puntos con 256 colores.

— Un paquete de diseño industrial y marcada.

— Otro tipo de software, como paquetes de gestión, hoja electrónica de cálculo, control de stocks, etc.

— Una consola alfanumérica para introducir los datos, y una impresora independiente para la creación de informes.

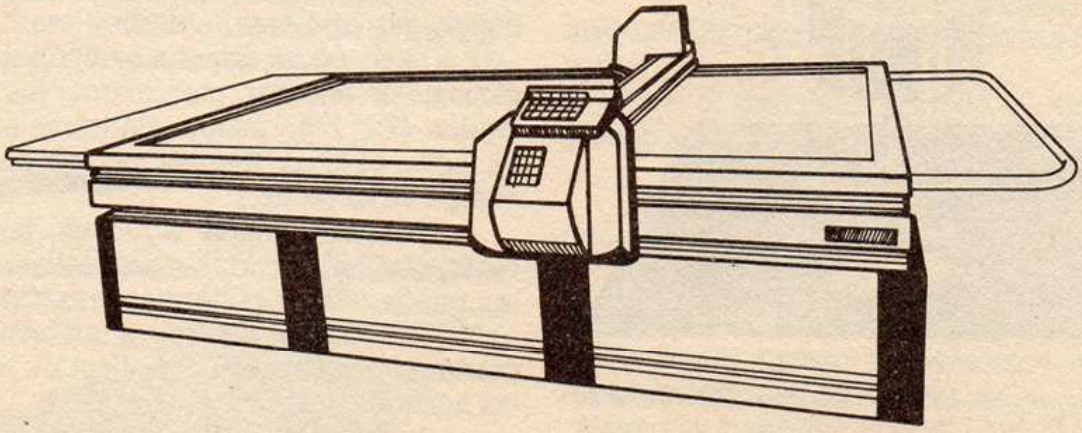
— Un plotter, muy rápido y de alta precisión.

— Multitarea. Pueden realizarse marcadas interactivamente, y al mismo tiempo hacer dibujos con el plotter, y digitalizar los patrones (ir tomando los datos de cada uno de los puntos inspeccionados).

— Red local. Pueden interconectarse varios sistemas a través de una red local.

— Corte automático. También es posible conectar «on line» cualquiera de los sistemas de corte automático disponibles.

El sistema de diseño ofrece una enorme libertad al diseñador. Utilizando un tablero electrónico y una pantalla, puede crear el modelo, y colorearlo con cualquiera de la enorme paleta de colores disponible. También tiene muchas más posibilidades (copia, introducción en el catálogo, etc.), pero no hablaremos de ello en este ejemplo, ya que se trata simplemente de la preparación del producto para ser "procesado" por la máquina automati-



Los ordenadores de la tercera generación son capaces de ejecutar varios programas simultáneamente. Para ello la memoria del ordenador está dividida en dos zonas: de "conversiones" y zona de "proceso".

zada programable, o robot propiamente dicho, en nuestro caso el sistema de marcada y corte.

Con el diseño ya creado, pasemos al corte.

El sistema que se seleccione debe cumplir las necesidades de la tarea que va a realizar. A la altura de cortar, es importante considerar la altura del material.

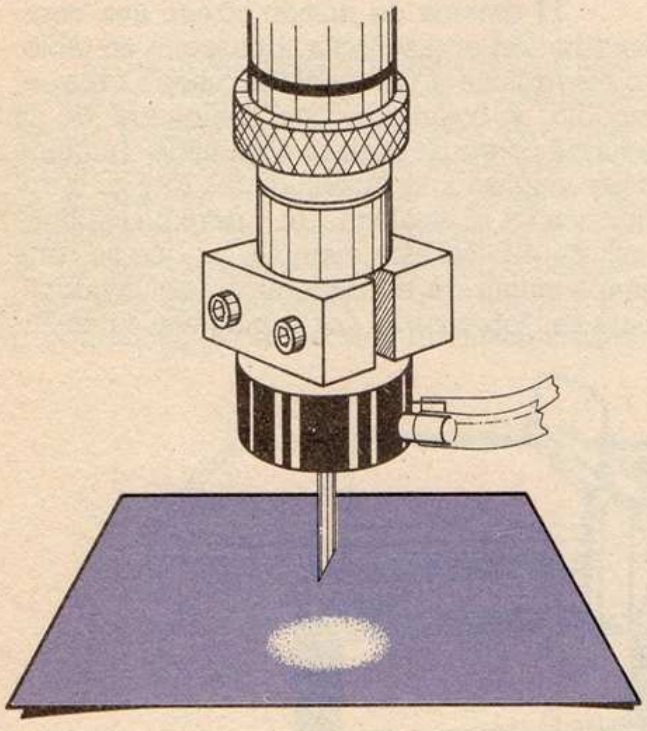
Así, pues, existen:

Máquinas de corte para alturas grandes.

Máquinas de corte para alturas moderadas.

Máquinas de corte para una sola capa de material, de alta velocidad.

Las máquinas de corte, en general, sin considerar especificaciones concretas sobre los distintos tipos de materiales a cortar, están controladas por un conjunto de microprocesadores en proceso paralelo. Además, disponen de un sistema de "aprendizaje" del robot, que le hace memorizar distintos datos y parámetros relacionados con el corte de determinados tipos de tejido. Entre estos datos, citemos la dureza, altura del molde (el molde está compuesto por el tejido doblado en una longitud determinada, el número de veces que se disponga), y la composición. También puede memorizar datos respecto de la marcada (dimensiones, proximidad entre piezas, etc.).



El teleprocesamiento permite al usuario de sistemas informáticos introducir los datos y recibir los resultados de un ordenador situado en cualquier otro lugar, siempre que exista la comunicación telefónica.

Toda la información respecto del corte se almacena en forma de fichero, y puede recuperarse en cualquier momento. Para ello, el equipo dispone de un microterminal donde se introducen las variaciones necesarias en los parámetros antes almacenados. Evidentemente, también puede reclamarse cualquier marcada, que puede pasar a ser cortada en ese mismo instante.

La máquina corta automáticamente "colchones" de tejido de gran altura con gran precisión, y a alta velocidad (es importante considerar que la cuchilla sufre presiones en su zona cortante, y esas presiones no son iguales en la zona situada más próxima a la articulación que en su extremo. Sin embargo, el corte debe ser todo el homogéneo y preciso).

El sistema de corrección de la flexión de la cuchilla asegura que todos los cortes sean idénticos en todo el espesor de la tela de cortar, ya que si no fuera así, la cuchilla al sufrir presiones, produciría que poco a poco fueran apareciendo diferencias cada vez mayores entre los primeros patrones situados en la parte superior de la tela a cortar, y los inferiores.

El afilado de la cuchilla es importante. El intervalo entre dos afilados puede regularse desde el panel de control, para adecuarse al tipo de material que se esté cortando (naturalmente, la cuchilla no se desgasta igual con un tipo de materiales que con otro). Como la precisión del corte depende también de la cuchilla, es importante que ésta esté siempre en condiciones óptimas, pero no por ello perder tiempo y material en operaciones de afilado inútiles.

El cabezal puede también ir provisto de un sistema de engrase de la cuchilla, que evita muchos problemas en materiales duros o difíciles, que pudieran fundirla.

La máquina dispone también de un mecanismo de taladrar, para marcar los distintos tejidos. (También puede taladrarse en caliente, para que las marcas permanezcan más tiempo en el tejido.)

Puede dibujarse siguiendo las marcas realizadas sobre cartón o papel. Para el trazado, puede utilizar bolígrafos o rotuladores.

El equipo incorpora un sistema de corrección automática del offset de las distintas herramientas.

Veamos las partes físicas que constituyen este sistema robot de marcada y corte:

El puente de corte

Es una estructura rígida y ligera, en la que van montados los motores de accionamiento de los ejes x e y (en el plano de la mesa). El puente se mueve por ésta por unas guías de acero (con los consiguientes rodamientos, mecanismos de piñón, cremallera, etcétera).

El puente transmite el movimiento al cabezal de corte a través de un mecanismo de polea y correa dentadas, y unas guías.

Sobre uno de los extremos del puente, cuya estructura es de chapa de aluminio, va montado el terminal de mando y comunicación, con un controlador de 8 líneas y 32 caracteres alfanuméricos. También puede moverse manualmente el sistema puente-cabezal en las direcciones x e y mediante un joystick. El equipo también dispone de unos pulsadores de emergencia muy evidentes a ambos lados.

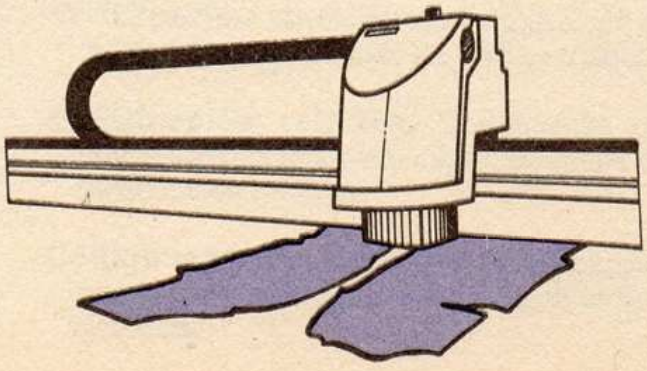
La alimentación de los motores y la comunicación con el controlador se realiza a través de unos cables que corren a lo largo de uno de los lados de la mesa, hasta llegar al lado del terminal del puente.

La mesa de corte

Es la mesa sobre la que se extiende el tejido a cortar. Soporta el puente y los cables de alimentación de motores y controlador.

En su parte superior, se encuentran tanto los sistemas de sujeción de tejido como las guías y cremalleras que permiten el movimiento del puente. El sistema de sujeción consiste en una colección de filamentos de nilon, unidos en su base, que permiten el corte de la cuchilla. Est sistema puede sacarse con facilidad para su limpieza y sustitución.

La mesa también dispone de un sistema de vacío, para aumentar la fijación del tejido a la base.



El controlador

En el controlador se encuentran los elementos encargados de controlar el funcionamiento de la máquina y las fuentes de alimentación del mismo.

La función del controlador es traducir los comandos de alto nivel del sistema en órdenes a motores y herramientas, y vigilar las señales de final de carrera en los eje x, y y z. El operador puede manipular el controlador a través del microterminal, que se encuentra en el puente, y a través de los mandos del armario.

El controlador está formado por una serie de microprocesadores, cada uno de los cuales realiza una tarea específica y controla una parte de la máquina. Las distintas partes se unen a través de buses paralelo de 8 bits.

El controlador está dividido en los módulos siguientes:

- Panel de mandos.
- Rack de lógica.
- Fuentes auxiliares.
- Fuentes lógicas.
- Fuentes de potencia.
- Rack de potencia.

Las labores de conversión de datos enviados desde el ordenador maestro al controlador y la generación de comandos de control de la trayectoria de la cuchilla y de su posición correcta en el cabezal se realizan con un software muy especializado. El afilado se realiza siguiendo las especificaciones del operador, que también puede optar por la utilización del taladro o del lapicero.

En el microterminal es donde aparecen los distintos tipos de mensajes del sistema que facilitan su manipulación. Se indican:

- Los parámetros de corte.
- Las piezas que se quedaron en la máquina en caso de parada forzada por algún error.
- El perímetro cortado, el tiempo empleado y la velocidad media de corte.
- Indicaciones de cuchilla gastada.
- Otros mensajes enviados por el ordenador de control.

Una de las misiones primordiales del controlador es detectar los fallos en el funcinamiento del sistema electrónico. Los errores más comunes son los siguientes:

- Posición irrecuperable en cualquiera de los tres ejes.

— Error en la memoria ROM o RAM del microprocesador maestro y de los microprocesadores esclavos.

— Error en la línea de transmisión.

— Error en la comunicación de la red de micros.

— Detección de falta de tensión o sobrecorriente en los motores.

Elementos auxiliares

Entre los elementos auxiliares tenemos los carros arrastramoldes para el traslado de los moldes desde la mesa de tendido, donde se forman los colchones hasta la de corte. Estos carros disponen de una estructura portante, con motores sincronizados mecánicamente y un accionamiento reductor montado directamente sobre uno de los grupos motores.

También disponen de un mecanismo sofisticado de sujeción del molde para poder arrastrarlo, y un panel de mandos.

El carro de transferencia del puente sirve para arrastrarlo junto con el cabezal de corte y los cables, desde una mesa a la otra. En la parte inferior lleva unas ruedas para mover el conjunto en dirección perpendicular a la mesa de corte. (El motor mueve una de las ruedas.) En la parte superior se encuentran unas guías y cremallera que coinciden con las de la mesa para recibir el puente y los cables. También dispone de otro pequeño movimento en la dirección longitudinal de la mesa de corte, para poder engancharse y desengancharse.

Los carros pueden regularse en altura y anchura, permitiendo pequeños ajustes de orientación.

Como posibilidades opcionales, este sistema puede ir provisto de un equipo de "matching" o casado para los tejidos con dibujo, cuadros o rayas. (Va provista de una cámara de televisión que "lee" el tejido, y una unidad de proceso de la imagen, que "decide" cómo casar las piezas, resolviendo todos los problemas.)

También puede optarse por máquinas veloces, que realizan el corte de una capa de tejido muy limpiamente, utilizando en lugar de la tradicional cuchilla un chorro de plasma a alta presión, o mediante rayo láser. Evidentemente, los sistemas de control, etc., son muy diferentes, aunque la concepción de la máquina es, esencialmente, la misma.

VOCABULARIO DE INFORMATICA

Código. *a)* Conjunto de reglamentaciones que indican cómo deben representarse los datos. Un ejemplo puede ser el Código ASCII (American Standard Code for Information Interchange - Código standard americano para el intercambio de información). *b)* En telecomunicaciones, conjunto de reglas y normas que deben cumplir las señales que representan los datos. *c)* En proceso de datos, sistema para representar datos o programas en una forma simbólica válida para que sea aceptada por un procesador. *d)* Puede referirse al código de una subrutina. *e)* Conjunto de elementos (como, por ejemplo, abreviaciones) que pueden utilizarse para representar los elementos de otro conjunto.

Código ampliado, carácter de. Carácter de control que se utiliza para indicar que uno o más de los valores correspondientes a los códigos siguientes deben interpretarse según otro código diferente.

Codificación absoluta. Codificación que utiliza instrucciones en código máquina, con direcciones absolutas. Sinónimo de codificación específica.

Codificación automática. Preparación del ordenador para rutinas en código máquina.

Codificación relativa. Codificación que utiliza instrucciones en código máquina con direcciones relativas.

Código máquina. Se refiere al lenguaje específico del microprocesador de un ordenador dado.

Color. En reconocimiento óptico de caracteres, característica espectral que depende de la reflectancia de la imagen, de la respuesta espectral del observador y de la composición de la luz incidente.

Columna. Disposición vertical de caracteres o expresiones. También puede designar la posición de un dígito.

Columna binaria. Perteneciente a una representación de datos en sistema binario en tarjetas perforadas. Las perforaciones se realizan en las distintas columnas. Por ejemplo, en una tarjeta de 12 filas, cada columna puede representar 12 bits consecutivos.

Combinatorio, elemento lógico. Dispositivo que dispone al menos de un canal de salida, y puede tener o no canales de entrada. Todos ellos se caracterizan por poseer estados discretos, de forma que el estado de cada canal de salida está determinado por los estados que tomen en ese momento los canales de entrada.

Comando. *a)* Señal de control. *b)* Instrucción en lenguaje máquina. *c)* También puede referirse no muy exactamente a un operador matemático o lógico.

Comandos, lenguaje de. Lenguaje fuente que consiste fundamentalmente en operadores de procedimientos, cada uno de los cuales puede llamar a una función para su ejecución.

Común, campo. Campo al que puede acce-

derse a través de una o más rutinas independientes.

Compilar. Paso de un programa escrito en un determinado lenguaje de programación a programa en código máquina. Para ello pueden utilizarse las estructuras lógicas del programa, o generar más de una instrucción en código máquina por cada instrucción simbólica.

Compilador. Programa que compila.

Complemento. Número derivado de otro específico que se obtiene sustrayéndolo de otro número especificado. Es frecuente representar los números negativos por sus complementarios.

Completo, arrastre. En adiciones paralelas, técnica que permite que se realicen todos los arrastres. En contraposición con arrastre parcial.

Computer. (Ver **Ordenador.**)

Concurrente. Cuando dos o más acontecimientos se llevan a cabo dentro del mismo intervalo de tiempo especificado. En contraposición con consecutivo, secuencial, simultáneo.

Condicional, salto. Salto producido cuando aparece determinado criterio.

Conector. En un diagrama de flujo medio para representar la convergencia de más de una línea de flujo en otra línea, o bien la divergencia de una línea de flujo en más líneas. También puede representar una ruptura en una única línea de flujo, para continuar en otra área. Dispositivo físico de conexión.

Conexión serie. (Ver **Serie, conexión en.**)

Conexión paralelo. (Ver **Paralelo, conexión en.**)

Consecutivo. Se refiere a dos acontecimientos secuenciales, que se producen sin la intervención de ningún otro acontecimiento. En contraposición con concurrente, secuencial y simultáneo.

Consola. Parte del ordenador que se utiliza como medio de comunicación entre el usuario y la máquina.

Constante. (Véase **Dirección constante.**)

Contraste. En reconocimiento de caracteres, la diferencia entre el color del carácter impreso y el fondo sobre el que éste está impreso.

Control numérico. Control automático de un proceso, realizado por un dispositivo que utiliza la totalidad o al menos una parte de los datos numéricos introducidos durante el proceso.

Control secuencial. Modo de operación del ordenador en el que las instrucciones se ejecutan según la secuencia definida implícitamente, hasta que se salta a otra secuencia mediante una instrucción de salto (*jump*).

Control, carácter de. Un carácter cuya aparición inicia, modifica o detiene una operación de control. Como ejemplo de este tipo de carácter tenemos el de control de vuelta del carro, o el de control de la transmisión de datos por una red de comunicaciones. Los caracteres de control se pueden almacenar para que sean efectivos en otro momento. En algunos casos, estos caracteres tienen una representación gráfica.

Control, operación de. Acción que lleva a cabo un determinado dispositivo, y cuyo resultado es el arranque o detención de un proceso determinado.

Control, panel de. Zona de la consola del ordenador en la que se encuentran los controles manuales.

Control, unidad de. En los ordenadores digitales, aquella parte que toma las instrucciones en la secuencia adecuada, interpreta cada instrucción y como resultado envía las señales adecuadas a las unidades que deben recibirlas.

Controlador. Sinónimo de Interface. Dispositivo electrónico que se utiliza para conectar periféricos a la unidad central de proceso.