

BASIC

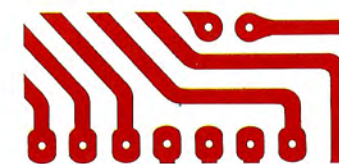
ENCICLOPEDIA DE LA INFORMÁTICA
MINIORDENADORES Y ORDENADORES PERSONALES





BASIC

ENCICLOPEDIA DE LA INFORMATICA.
MINIORDENADORES Y ORDENADORES PERSONALES

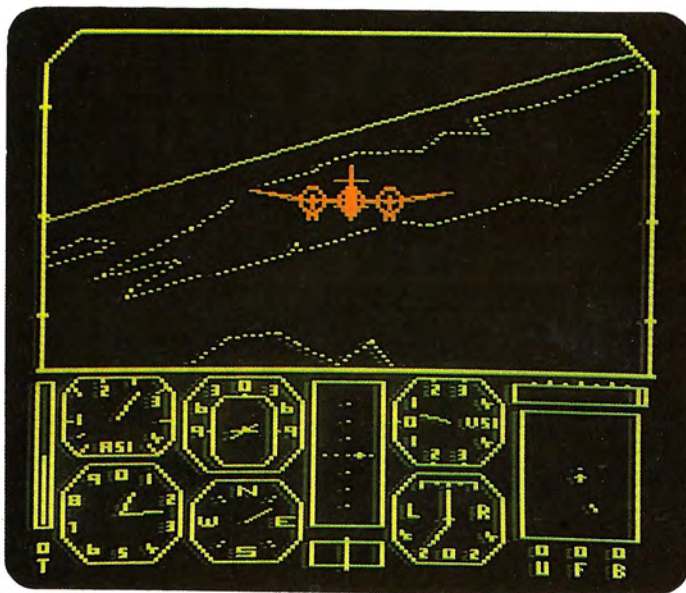


ENCICLOPEDIA DE LA INFORMATICA.
MINIORDENADORES
Y ORDENADORES PERSONALES

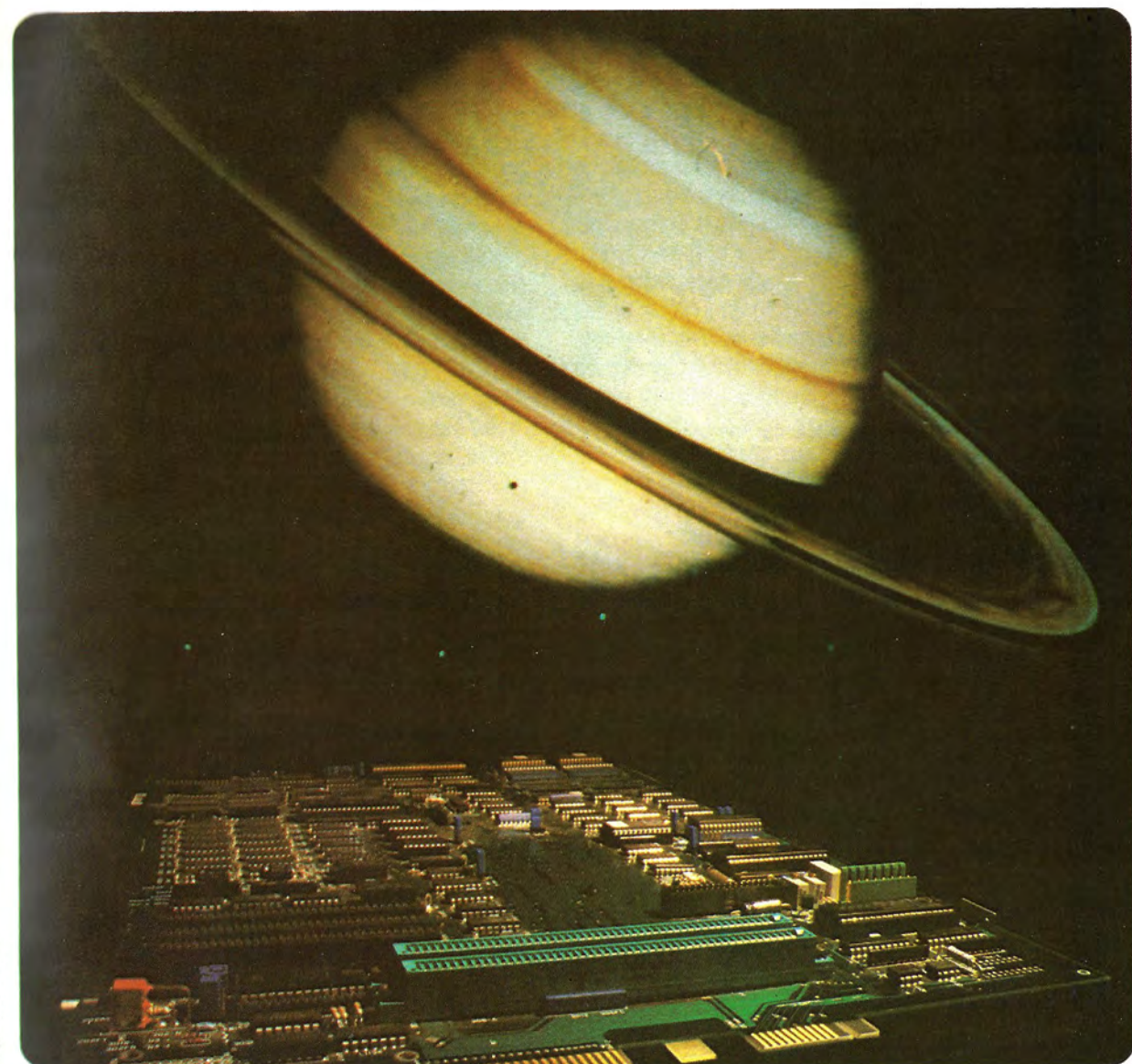
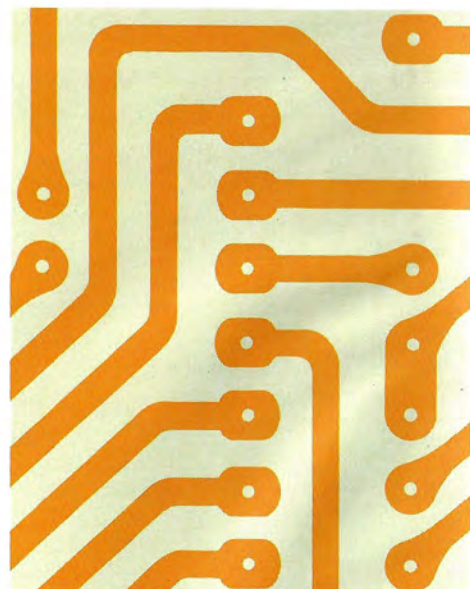
BASIC



Marka-Zeifa



Marka-Zeifa



C. Pizzoni/Marka

BASIC

ENCICLOPEDIA DE LA INFORMATICA. MINIORDENADORES Y ORDENADORES PERSONALES

Presidente

José Manuel Lara

Director Ejecutivo

Jesús Domingo

Dirección editorial

R.B.A., Proyectos Editoriales, S.A.

Dirección técnica

Sante Senni

Programas de **Enrico Calcara**

Con el asesoramiento de la Sociedad **E.G.S.**

REDACCION

Dirección: Gabriella Costarelli

Redactor jefe: Marcella Marcaccini

Secretaría de redacción: Giulia Abriani, Maria Pierantozzi

Redacción: Ugo Spezia

Corrección: Maria Albergo, Laura Salvini, Graziella Tassi

Recopilación material gráfico: Carla Bertini, Rossella Pozza

Producción: Piergiorgio Palma

Secretaría de producción: Maria Rita Ciucci

Diseño y dirección artística: Vittorio Antinori

Jefe estudio gráfico: Roberto Sed

Compaginación: Alberto Berni, Riccardo Catani, Patrizia Fazio

Dibujos: R. Giorgini, R. Lazzarini, G. Mazzoleni, R. Mazzoni

Traducción: Carlo Frabetti

Diseño cubierta: Nestlé Soulé

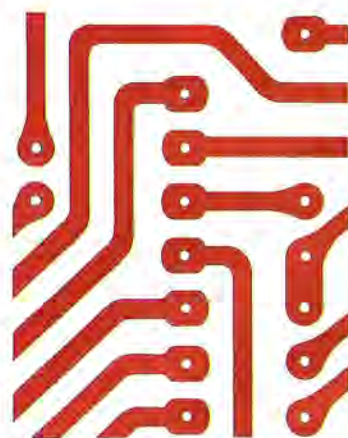
Jefe de Producción: Ricardo Prats

© 1983 Ediciones Forum, S.A., Córcega 273, Barcelona-8
© Armando Curcio Editor, Roma. Reservados todos los derechos.
Título original: BASIC
Enciclopedia dell'informatica dei mini e personal computer
Prohibida la reproducción por cualquier medio sin el permiso
escrito del editor.

Composición electrónica: ITC-Fototipo, Barcelona-Madrid.
Imprime: CAYFOSA - Carretera de Caidas, Km. 3,7
Santa Perpetua de Mogoda (Barcelona)

Depósito Legal: B. 37.099/83
ISBN (Obra completa): 84-7574-040-5
ISBN (Volumen VI): 84-7574-377-3
ISBN (Fascículos): 84-7574-044-8
Impreso en España - Printed in Spain

El editor agradece la colaboración de:
Alfa Romeo, Buffetti Data, Commodore Italiana, CPT Italia, Creazioni Walt Disney,
Data General, Digital, Doxa, Elsas, Ericsson, Facit Data Products, Ferrari, FIAT,
Harden Italia, Hengstler Italia, Hewlett Packard, IBM, Intema, IRET Informática,
Italcable, Lancia, Litton BEI, MEE, MSI Data Italia, Olivetti, Perkin-Elmer, Plessey
Trading, Prime Italia, Rank Xerox, Rhône-Poulenc Italia, Sanco Ibex Italia, Sarin,
Seica Elettronica, Selenia, Sperry, SIP, Telespazio.



La parte inicial del programa (líneas 10 a 40) se dedica a la implantación de los parámetros característicos del monitor empleado. Los valores $N = 5$, $S = 153$, $E = 265$, $O = 15$ delimitan el área útil (en puntos de pantalla), mientras que los valores $X0 = 140$ e $Y0 = 80$ representan el origen respecto al cual se refieren las coordenadas (estos valores se refieren al Siprel 2010. La nomenclatura utilizada (N, S, E, O) se refiere a los puntos cardinales (N = Norte, S = Sur, etc.) e identifica la «ventana» utilizada para la presentación del gráfico. Las instrucciones 140, 150 y 160 calculan los valores XA , YA y XB , YB en puntos de pantalla. Con respecto al diagrama de flujo, las fórmulas varían en las cantidades $X0$, $Y0$, puesto que en el caso general (diagrama de flujo), las coordenadas de cada punto de la curva se refieren al origen O de coordenadas $X0 = 0$, $Y0 = 0$, mientras que en el listado, el origen se ha fijado en 140, 80, que permite una presentación centrada en la pantalla. Además, obsérvese que la coordenada Y está calculada como diferencia entre el origen y el valor dado de la ecuación de la recta. Esto es debido a la orientación del eje Y, que en la máquina particular utilizada empieza arriba a la izquierda. El cálculo del paso de presentación se inserta en la línea 110, mientras que el bucle se realiza entre las líneas 120 y 190. La línea 160 controla que los valores de las coordenadas (en puntos de pantalla) no superen los valores máximos (línea 20). Obsérvese que al final del bucle (línea 190), para brevedad, se ha omitido el índice (NEXT en lugar de NEXT I) y el modo particular adoptado en la línea 230 para reactivar el programa. Esta sintaxis no puede utilizarse en todas las máquinas; en estos casos es necesario completar la línea 190 y escribir la 230 en la forma

```
IF SS = "S" THEN GOTO 50
```

Además, para transportar el programa a otras máquinas es necesario sustituir las siguientes instrucciones:

HOME borra el monitor y posiciona el cursor arriba a la izquierda (instrucción 200)

VTABn tabulación vertical con paro en la línea n (instrucción 210)

H PLOT traza un segmento entre los dos puntos con las coordenadas especificadas en la instrucción (instrucción 1010)

HGR activa el modo gráfico (instrucción 1000)

El programa presentado no es el más racional para trazar una recta: todas las instrucciones comprendidas entre la 100 y la 190 pueden estar constituidas por la sola instrucción de trazado de un segmento entre los puntos de coordenadas XI , YI y XF , YF . En este ejemplo se ha adoptado el bucle para generalizar el programa, que de esta manera permite presentar una función cualquiera diferente de la recta; basta con modificar la línea 40 introduciendo la definición de la función deseada, aunque también hacen falta otros controles y otros cálculos, que se presentarán más adelante.

Presentación de los ejes cartesianos

La presentación del gráfico de una función requiere también la visualización de los ejes cartesianos a los que está referida. En la figura de las páginas 1442 y 1443 se ha representado el diagrama de flujo de una subrutina que dibuja de manera parametrizada un par de ejes cartesianos; el listado correspondiente está incluido en el programa de las páginas 1446 a 1448. Los parámetros a proporcionar en la entrada tienen los mismos significados vistos anteriormente. La rutina traza los ejes como dos segmentos, perpendiculares entre sí, que se encuentran en el punto $X0$, $Y0$, elegido como origen (ver la página 1444). En los extremos de los ejes dibuja las flechas que indican su orientación, y al final presenta las divisiones en cada eje.

El símbolo «flecha» se obtiene trazando algunos segmentos paralelos de longitud decreciente (instrucciones 1030 a 1040 del listado), como se indica en la figura de la página 1444. Podría haberse utilizado un triángulo, pero el resultado estético sería menos satisfactorio. Sin embargo, en la figura se ha ilustrado también esta solución. Indicando con N el máximo valor positivo de la ordenada Y, las instrucciones a introducir se reducen al trazado de tres segmentos

	de	a
Segmento 1	$X0 - 1, N$	$X0 + 1, N$
Segmento 2	$X0 + 1, N$	$X0, N - 3$
Segmento 3	$X0, N - 3$	$X0 - 1, N$

y pueden ser incluidos en una sola línea del programa

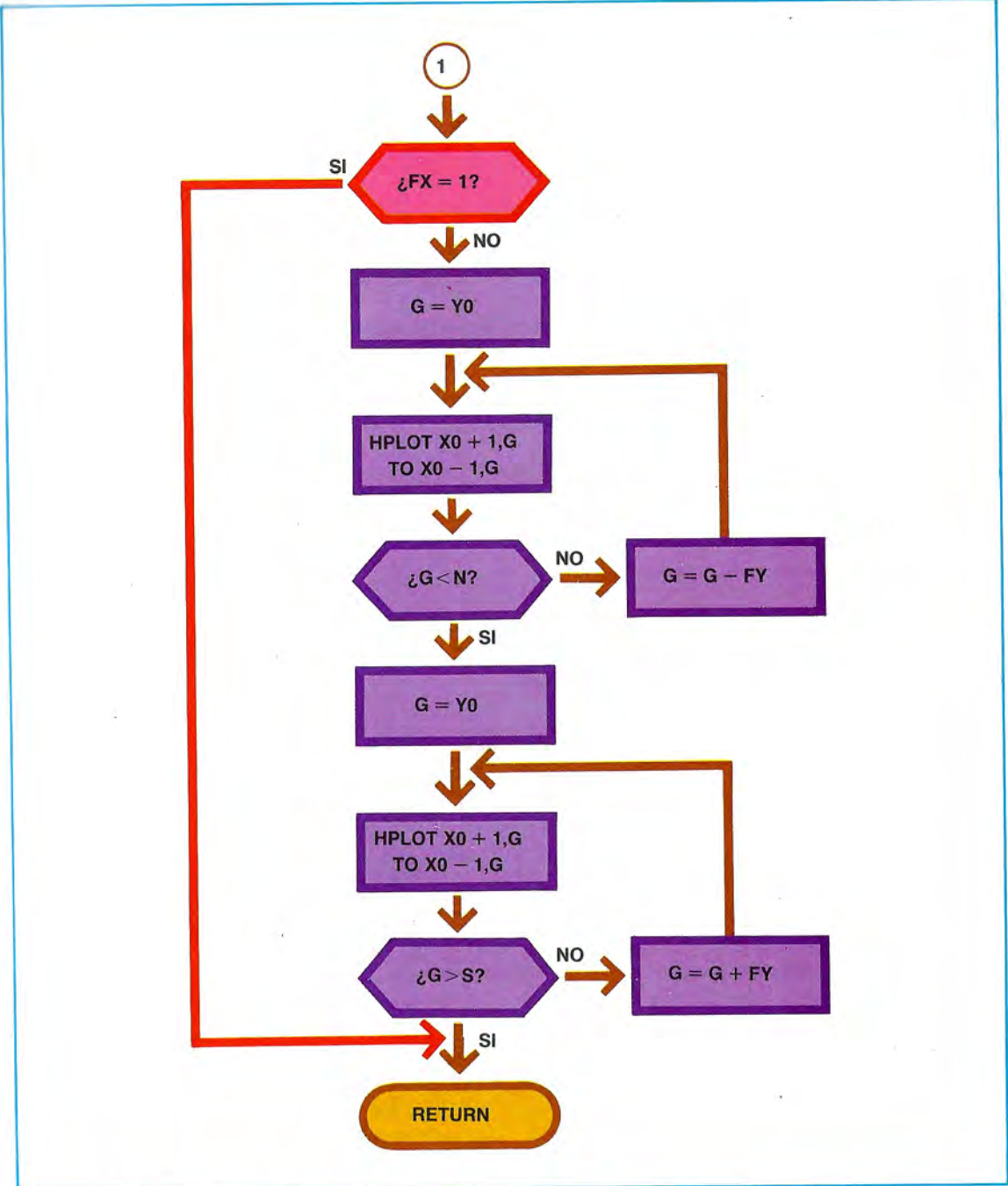
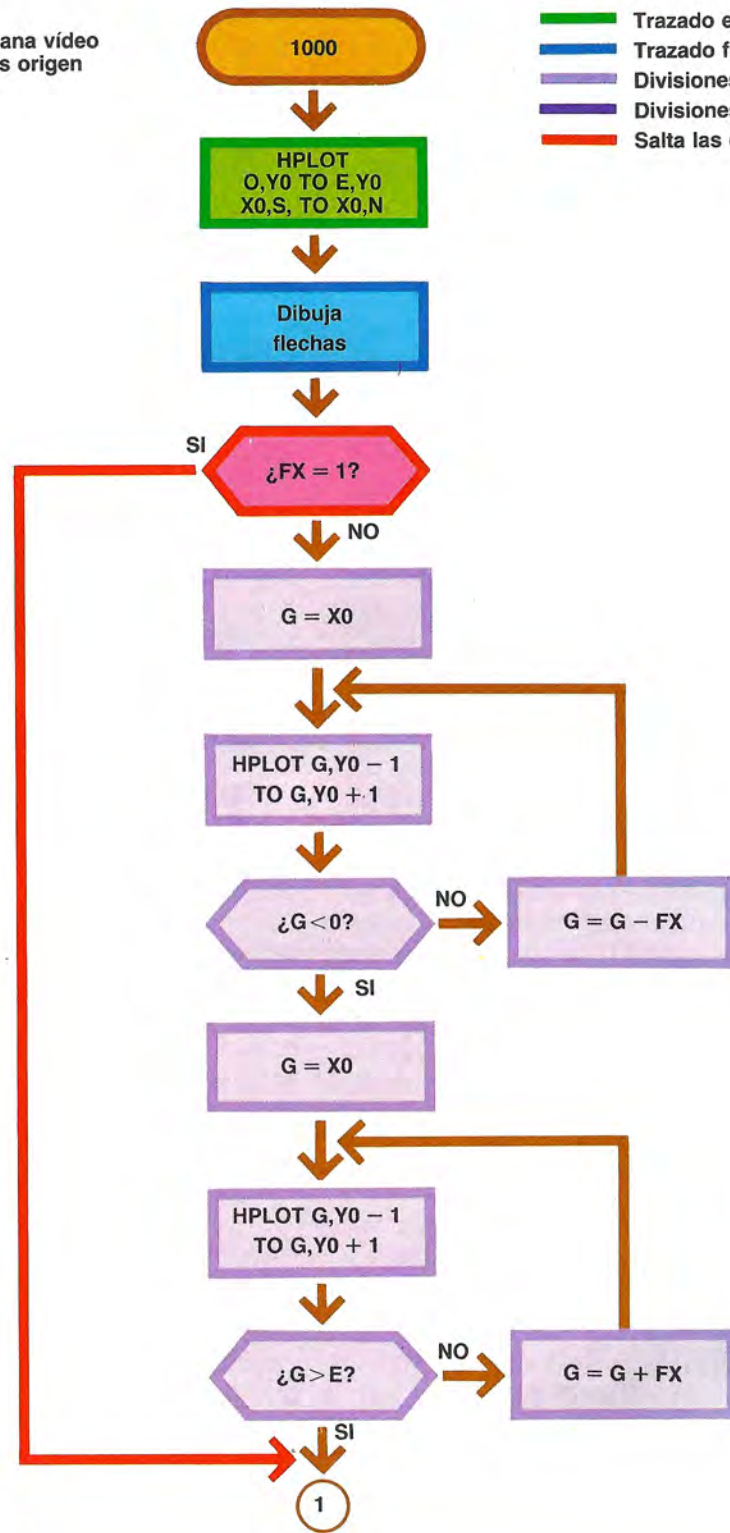
```
H PLOT X0 - 1, N TO X0 + 1, N TO X0, N - 3 TO X0 - 1, N
```

Para los otros tres extremos de los ejes (S, E, O), las instrucciones son análogas.

DIAGRAMA DE FLUJO DE LA SUBROUTINA PARA DIBUJAR LOS EJES

Entradas
 N,S,E,O Límites ventana vídeo
 X0,Y0 Coordenadas origen
 FX,FY Escala ejes

- █ Trazado ejes
- █ Trazado flechas
- █ Divisiones eje x
- █ Divisiones eje y
- █ Salta las divisiones



Trazado de la recta que pasa por dos puntos dados

En muchas aplicaciones gráficas es útil disponer de una rutina que pueda trazar la ecuación de una recta que pasa por dos puntos de determinadas coordenadas. Ya sabemos que la ecuación de una recta general puede escribirse en la forma: $Y = A * X + B$; para obtener la recta particular que pasa por los dos puntos dados

de coordenadas $X1, Y1$ y $X2, Y2$, pueden aplicarse las siguientes fórmulas

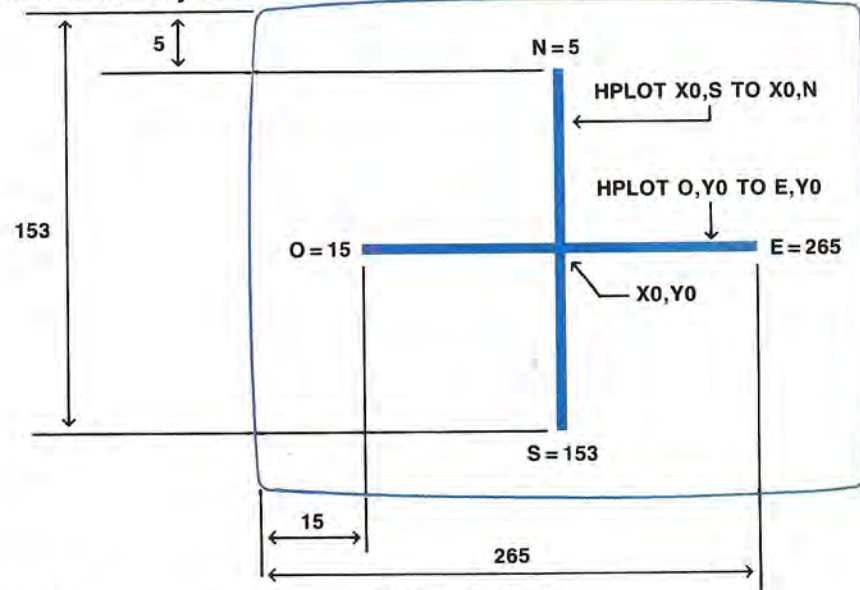
$$A = \frac{Y2 - Y1}{X2 - X1}$$

$$B = \frac{(X1 * X2) - (Y2 * X1)}{X2 - X1}$$

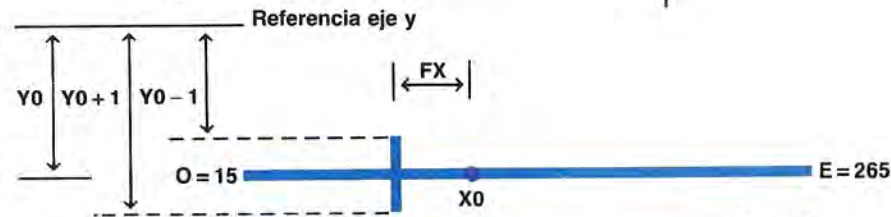
Por ejemplo, si los puntos son $X1 = 2, Y1 = 2.5$ y $X2 = 5, Y2 = 4$, se tiene:

TRAZADO DE LOS EJES CARTESIANOS

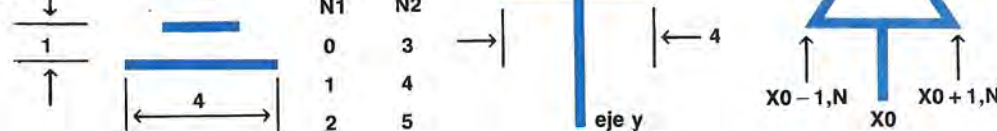
Posicionamiento de los ejes



Subdivisiones



Flechas



$$A = \frac{4 - 2.5}{5 - 2} = \frac{1.5}{3} = 0.5$$

$$B = \frac{(2.5 * 5) - (4 * 2)}{5 - 2} = \frac{12.5 - 8}{3} = \frac{4.5}{3} = 1.5$$

Por tanto, la recta buscada tiene la ecuación

$$Y = 0.5 * X + 1.5$$

Las fórmulas ilustradas pueden utilizarse para realizar una rutina que pueda visualizar una recta que pasa por los puntos de coordenadas conocidas, introducidas por ejemplo por teclado. El diagrama de flujo de principio de la rutina

puede verse en la página siguiente, y el listado en las páginas 1446 y 1447. El programa pide las coordenadas del primer y del segundo punto (líneas 80 a 110), y dibuja, en correspondencia con las coordenadas introducidas, dos símbolos (en este caso dos cuadrados, subrutina 3000); después calcula los valores de A y B (líneas 120 a 130) y traza la recta (bucle de la línea 170 a la 240).

Esta solución sólo se presenta para ilustrar un método que utilizaremos más adelante; para obtener el mismo resultado habría bastado la instrucción de trazado del segmento que une los

DETERMINACION DE LA ECUACION Y TRAZADO DE UNA RECTA QUE PASA POR DOS PUNTOS DADOS



- Tabla de los símbolos: SX(-), SY(-)
- Límites de la pantalla: N,S,E,O
- Coordenadas origen: X0, Y0
- Factores de escala: FX,FY
- Números de puntos: NP
- Factores de escala de los símbolos: S1,S2

Esta rutina se inserta sólo para evidenciar los dos puntos

dos puntos dados (HPLLOT X1,Y1 TO X2,Y2), pero de esta manera no se hubiera dispuesto de dos coeficientes A y B. La recta se habría presentado con un segmento de unión de los puntos X1,Y1 y X2,Y2, sin que el programa conociera la forma analítica, o sea la ecuación $Y = A * X + B$. En cambio, si la recta se indica matemáticamente, la ecuación puede utilizarse para otros procesos, como por ejemplo para el cálculo de puntos intermedios entre los dados. El listado muestra el uso de una tabla de los desplazamientos que sirve para generar un símbolo en un punto cualquiera de la pantalla. En

las líneas 20 y 30 se memorizan los valores de las coordenadas que determinan los lados del símbolo (en este caso, un cuadrado); para trazarlo en un punto es necesario sumar estos valores a las coordenadas del punto en cuestión. En la subrutina 3000 aparecen además los coeficientes S1 y S2, mediante los cuales se puede variar la dimensión de cada segmento. En el listado se han hecho ambos iguales a 2 (línea 50), y así se genera un cuadrado. El bucle que dibuja los primeros tres lados del símbolo se realiza entre la línea 3050 y la 3090; el último lado se dibuja con la línea 3100.

TRAZADO DE UNA RECTA QUE PASA POR DOS PUNTOS

Versión Apple y compatibles

```

10 HOME
20 FOR J = 1 TO 4
  READ SX(J),SY(J)
NEXT J
30 DATA -1,1,1,1,1,-1,-1,-1
40 N = 5
   S = 153
   E = 265
   O = 15
50 XO = 140
   YO = 80
   FX = 5
   FY = 5
   NP = 10
   S1 = 2
   S2 = 2
60 GOSUB 1000
70 DEF FN R(X) = A * X + B
80 VTAB 23
  INPUT "COORDENADAS PRIMER PUNTO: ";
    X1,Y1
90 XS = X1 * FX + XO
   YS = YO - Y1 * FY
  GOSUB 3000
100 HOME
110 VTAB 23
  INPUT "COORDENADAS SEGUNDO PUNTO: ";
    X2,Y2
120 A = (Y2 - Y1) / (X2 - X1)
130 B = ((Y1 - X2) - (Y2 * X1)) / (X2
  - X1)
140 XS = X2 * FX + XO
   YS = YO - Y2 * FY
  GOSUB 3000
150 P = (X2 - X1) / NP
160 X = X1
   Y = FN R(X)
170 FOR XX = X1 + P TO X2 STEP P
180 YY = FN R (XX)
190 XA = XO + FX * X
   YA = YO - FY * Y
200 XB = XO + FX * XX
   YB = YO - FY * YY
210 IF XA > 265 OR XA < 15 OR XB > 26
   5 OR XB < 15 OR YA > 153 OR YA
   < 5 OR YB > 153 OR YB < 5
   THEN 230
220 HPLOT XA,YA TO XB,YB
230 X = XX
   Y = YY
240 NEXT
250 HOME
260 VTAB 23
270 INPUT "CONTINUO? (S/N) ".$S
280 IF S$ = "S" THEN RUN
290 END
297 REM -----
298 REM DIBUJA EJES
299 REM -----
1000 HGR
   HCOLOR= 3
1010 HPLOT 0 - 15,N - 5 TO E + 14,N
   - 5 TO E + 14,5 + 3 TO 0 - 15,5
   + 3 TO 0 - 15,N - 5
1020 HPLOT 0,YO TO E,YO
   HPLOT XO,S TO XO,N
1030 N2 = N
   FOR N1 = 2 TO 0 STEP - 1

```

```

   HPLOT XO - N1,N2 TO XO + N1,N2
   N2 = N2 -1
   NEXT N1
1040 N2 = E
   FOR N1 = 2 TO 0 STEP - 1
   HPLOT N2,YO - N1 TO N2,YO + N1
   N2 = N2 + 1
   NEXT N1
1050 HPLOT E + 6,YO - 2 TO E + 11,YO
   + 3
   HPLOT E + 11,YO - 2 TO E + 6,YO
   + 3
1060 HPLOT XO - 10,N - 2 TO XO - 7,N
   + 1
   HPLOT XO - 4,N - 2 TO XO - 10,N
   + 3
1070 IF FX = 1 THEN 1140
1080 FOR G = XO TO 0 STEP - FX
1090 HPLOT G,YO + 1 TO G,YO - 1
1100 NEXT G
1110 FOR G = XO TO E STEP FX
1120 HPLOT G,YO + 1 TO G,YO - 1
1130 NEXT G
1140 IF FY = 1 THEN 1210
1150 FOR G = YO TO N STEP - FY
1160 HPLOT XO + 1,G TO XO - 1,G
1170 NEXT G
1180 FOR G = YO TO S STEP FY
1190 HPLOT XO + 1,G TO XO - 1,G
1200 NEXT G
1210 RETURN
3000 REM -----
3010 REM DIBUJA SIMBOLOS
3020 REM -----
3030 XA = SX(1) * S1 + XS
   YA = SY(1) * S2 + YS
3040 XC = XA
   YC = YA
3050 FOR I = 2 TO 4
3060 XB = SX(I) * S1 + XS
   YB = SY(I) * S2 + YS
3070 HPLOT XA,YA TO XB,YB
3080 XA = XB
   YA = YB
3090 NEXT I
3100 HPLOT XB,YB TO XC,YC
3110 RETURN

```

Versión Olivetti M20

```

10 CLEAR
20 NP = 10
30 W2 = WINDOW(2,40)
40 DEF FN R(X) = A * X + B
50 CLS%2
60 LINE%2(0,0) - (511,255),3,B
70 CLS
80 SCALE%2, - 50,50, - 50,50
90 GOSUB 270
100 INPUT "Primer punto":X1,Y1
110 XA = X1
   YA = Y1
   GOSUB 3000
120 INPUT "Segundo punto":X2,Y2
130 XA = X2
   YA = Y2
   GOSUB 3000
140 P = (X2 - X1) / NP

```



```

150 A = (Y2 - Y1) / (X2 - X1)
160 B = ((Y1 * X2) - (Y2 * X1)) / (X2
    - X1)

170 X = X1
    :Y = FN R(X)
180 FOR XX = X1 + P TO X2 STEP P
190 YY = FN R(XX)
200 LINE%2(X,Y) - (XX,YY),2
210 X = XX
    Y = YY
220 NEXT
230 INPUT "Continuo (S/N) ";A$
240 IF A$ = "S" THEN 10
250 CLEAR
    END

270 REM **** Dibuja ejes ****
280 LINE%2(0, - 50) - (0,50)
290 LINE%2(- 50,0) - (50,0)
300 FOR Q = 0 TO 49 STEP 2
310 LINE%2(- .5,Q) - (.5,Q)
320 LINE%2(Q, - .5) - (Q,.5)
330 NEXT Q
340 FOR Q = 0 TO - 49 STEP -2
350 LINE%2(- .5,Q) - (.5,Q)
360 LINE%2(Q, - .5) - (Q,.5)
370 NEXT Q
380 RETURN

3000 REM **** Dibuja símbolos ****
3010 LINE%2(XA - 2,YA -2) - (XA + 2,Y
    A + 2),2,B
3020 RETURN

```

Para transferir el programa a otras máquinas (la forma presentada vale para los sistemas Sipel, Apple y compatibles), deben modificarse las instrucciones VTAB, HPLOT, HCOLOR, HGR y los controles de la línea 210, sustituyendo los valores numéricos por los correspondientes a la pantalla empleada.

A título de ejemplo, en la página anterior y arriba se ha representado el mismo listado en la versión para el Olivetti M20.

Regresión lineal de mínimos cuadrados

El análisis de un fenómeno cualquiera consiste en tener bajo observación los valores que asumen las diferentes variables que interesan al fenómeno durante su evolución.

El caso más sencillo se tiene cuando las variables son dos (variable independiente y variable dependiente); el conjunto de los valores que asume la variable dependiente al variar la variable independiente, oportunamente representado, muestra el proceso del fenómeno.

Por ejemplo, supongamos que cinco observaciones hayan dado los siguientes valores (X e Y son las dos variables interesadas):

Observación	X	Y
a	2.5	3
b	4	3.5
c	5	5
d	7	4.5
e	9	6

El análisis de la tabla no proporciona ningún elemento de juicio, ni permite prever cuál podrá ser el valor de Y correspondiente a un valor de X no observado. Un primer método que permite obtener estas informaciones, y que ya se ha presentado, consiste en interpolar linealmente entre cada par de valores medidos. De esta manera, el proceso real se asimila a una serie de segmentos que unen los puntos a, b, c, d, e. El gráfico que se obtiene (figura de la página siguiente) puede estar muy alejado de la realidad, y los valores logrados con ellos pueden contener errores incluso notables. Adoptando este método, las ecuaciones que representan el fenómeno son las de las cuatro rectas que unen de dos en dos los puntos anotados. La ecuación de cada recta puede obtenerse con el método ilustrado en el párrafo anterior.

El segundo método, que en ciertas condiciones da resultados más adecuados, es la «regresión lineal de mínimos cuadrados».

Supongamos por ahora que el proceso del fenómeno observado pueda estar representado con una recta. Realizando una serie de observaciones sobre las variables interesadas se tendrán valores no del todo pertenecientes a la recta que representa matemáticamente el fenómeno. Es decir, también si la ley que regula el fenómeno fuese representada por la recta de trazos de la figura de abajo, los valores medidos podrían ser los puntos a, b, ..., y, que, salvo uno, no caerían exactamente sobre la recta. Esta dispersión depende de numerosos factores. Si la observación se refiere a un fenómeno físico, las causas de la dispersión de los valores deben buscarse en los errores instrumentales y en los accidentales. Cualquier instrumento, por perfecto que sea, no proporciona el valor exacto de

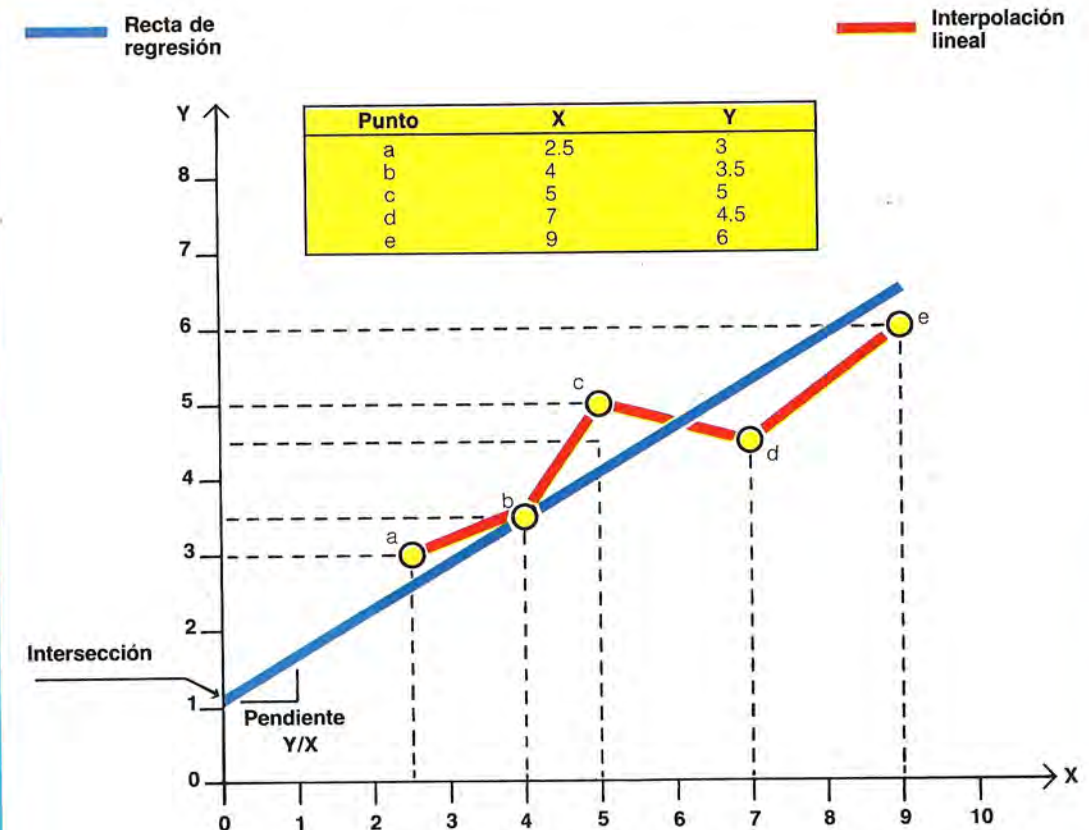
una magnitud, sino un valor tanto más próximo al mismo cuanto mejor es la calidad del instrumento. Una segunda causa de error es la influencia de factores externos accidentales, por ejemplo vibraciones mecánicas, variaciones de temperatura o de humedad, etc.

Estos errores, al combinarse, generan una dispersión de los valores observados, y sólo casualmente pueden anularse. Por lo demás, la verificación de esta última situación particular no puede preverse y por tanto, si en un cierto instante los valores no tienen error, no existe modo de evidenciarlo y los datos deben considerarse afectados de error en cualquier caso.

Por tanto, en general, al analizar un fenómeno, los valores observados se presentan dispersos alrededor de la función que la representa globalmente, que a su vez no es conocida.

El método de los mínimos cuadrados permite obtener la función que mejor se aproxima al fe-

PROCESO DE UN FENOMENO OBTENIDO CON CINCO OBSERVACIONES



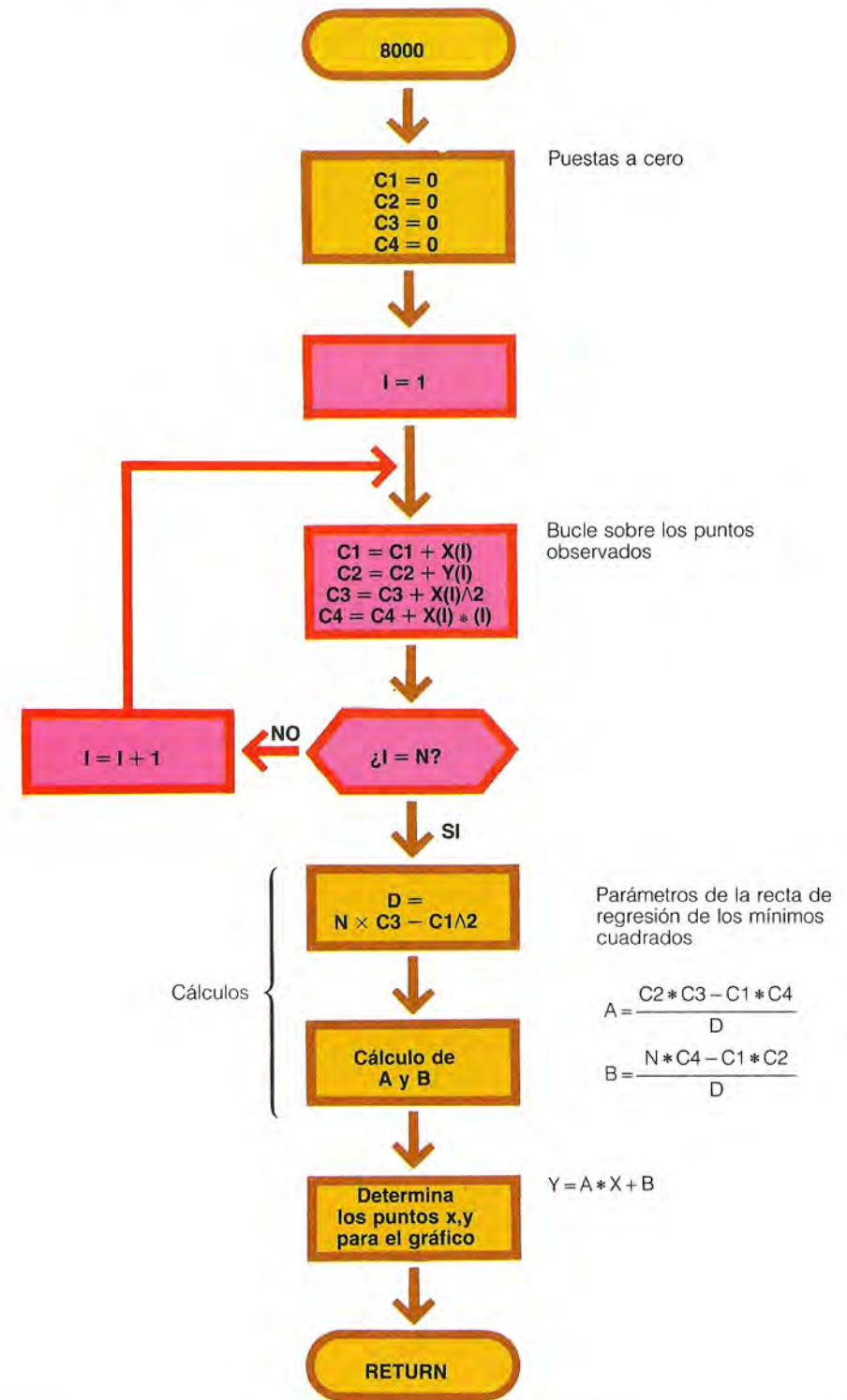


Ejemplo de aplicación gráfica en un Olivetti M20.

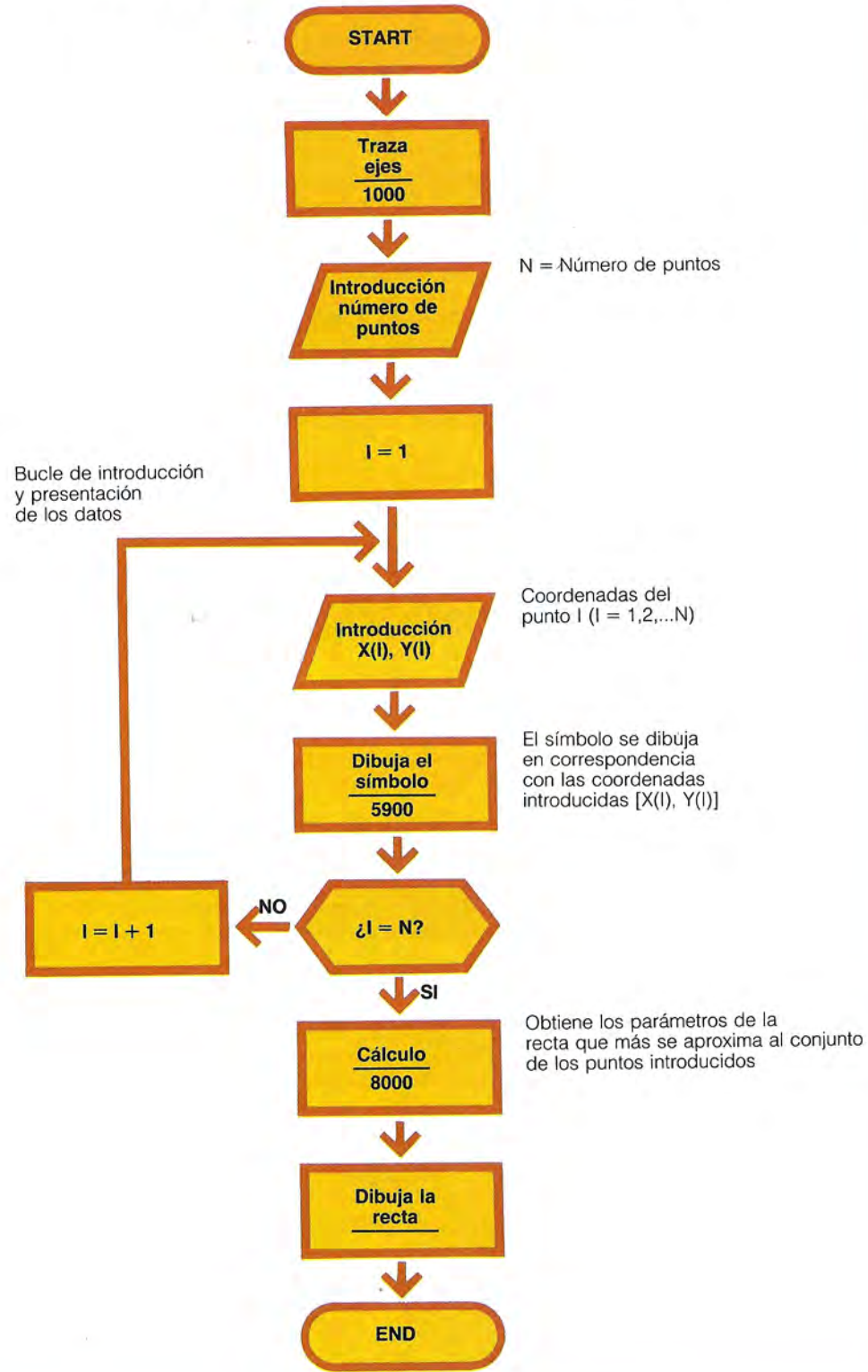
nómeno examinado. Las fórmulas a aplicar se derivan de un complejo análisis, y dependen del proceso supuesto para la función. El caso más sencillo es el de una recta (regresión lineal de mínimos cuadrados), y las fórmulas a aplicar en este caso para obtener la ecuación de la recta se indican en la subrutina 8000 de la figura de la página siguiente. Las variables A y B son los coeficientes de la ecuación de la recta que representa el fenómeno, y se calculan con los valores X (.) e Y (.) observados. N es el número de observaciones, mientras que las variables C1, C2, C3, C4 y D sirven de apoyo para los cálculos intermedios. Para utilizar la rutina deben memorizarse en X(.) e Y(.) las mediciones y las observaciones, y en N su número; en la salida se obtienen los coeficientes A y B de la recta que por hipótesis representa el fenómeno. En la figura de la página 1452 se ha representado el diagrama de flujo de principio. El primer paso consiste en trazar los ejes cartesianos; después, el programa pide el número de puntos (N) y los valores de las coordenadas de los pun-

tos correspondientes (bucle con índice I). Para cada par de coordenadas, la subrutina 5900 dibuja un cuadrado en el punto correspondiente al valor observado. Después del bucle de introducción, se llama la subrutina 8000 (figura de la página siguiente), que realiza el cálculo de los coeficientes A y B. Al final se tiene la presentación de la recta así identificada. El desarrollo del programa, en realidad es mucho más complejo de lo que puede verse en la figura de la página 1452. Las principales dificultades a superar se deben a los factores de escala adoptados. Durante la fase de introducción pueden presentarse valores de X o de Y que en la escala adoptada salen de los límites de la pantalla. Análogamente, después de haber calculado la ecuación de la recta ($Y = B * X + A$), representándola entre los valores $X = XN$ y $X = XM$ (respectivamente mínimo y máximo en el eje X), puede suceder que los valores correspondientes de la Y superen los límites de la pantalla. El primer control (límite en los valores X, Y) se realiza después de cada introducción, mientras que la comprobación en la recta se incluye en la subrutina 8000 (cuyo listado difiere del diagrama de flujo presentado). En las figuras de las páginas 1453 a 1455 se ha representado el diagrama de flujo detallado. El programa presenta algunas complicaciones debidas a la necesidad de prever un cambio de escala automático. Inicialmente, el origen de los ejes está en $X0 = 140$ e $Y0 = 80$, con un factor de escala $F = 0.7$. En cada introducción sucesiva se activa un bucle de control para comprobar que los nuevos puntos todavía pueden representarse en la escala elegida por omisión; si esto no es posible debe calcular un nuevo valor del factor de escala y volver a dibujar completamente el gráfico. La lógica expuesta se obtiene con una serie de controles y de flags. Para cada entrada vuelven a calcularse $[X(I), Y(I)]$ los valores máximo y mínimo sobre los dos ejes (XN, XM e YN, YM , líneas 270 a 360) y las amplitudes de los intervalos (DX, DY), entre los cuales se elige la mayor. En base a este valor vuelve a calcularse el factor de escala F, que debe ser un valor menor que 1; en caso contrario se produce un error y los datos no son válidos: el proceso se interrumpe y el programa vuelve a empezar desde el principio (con la instrucción RUN después del test $F < 1$). Determinado el factor de escala, se pasa al cálculo de las nuevas coordenadas $X0, Y0$ del ori-

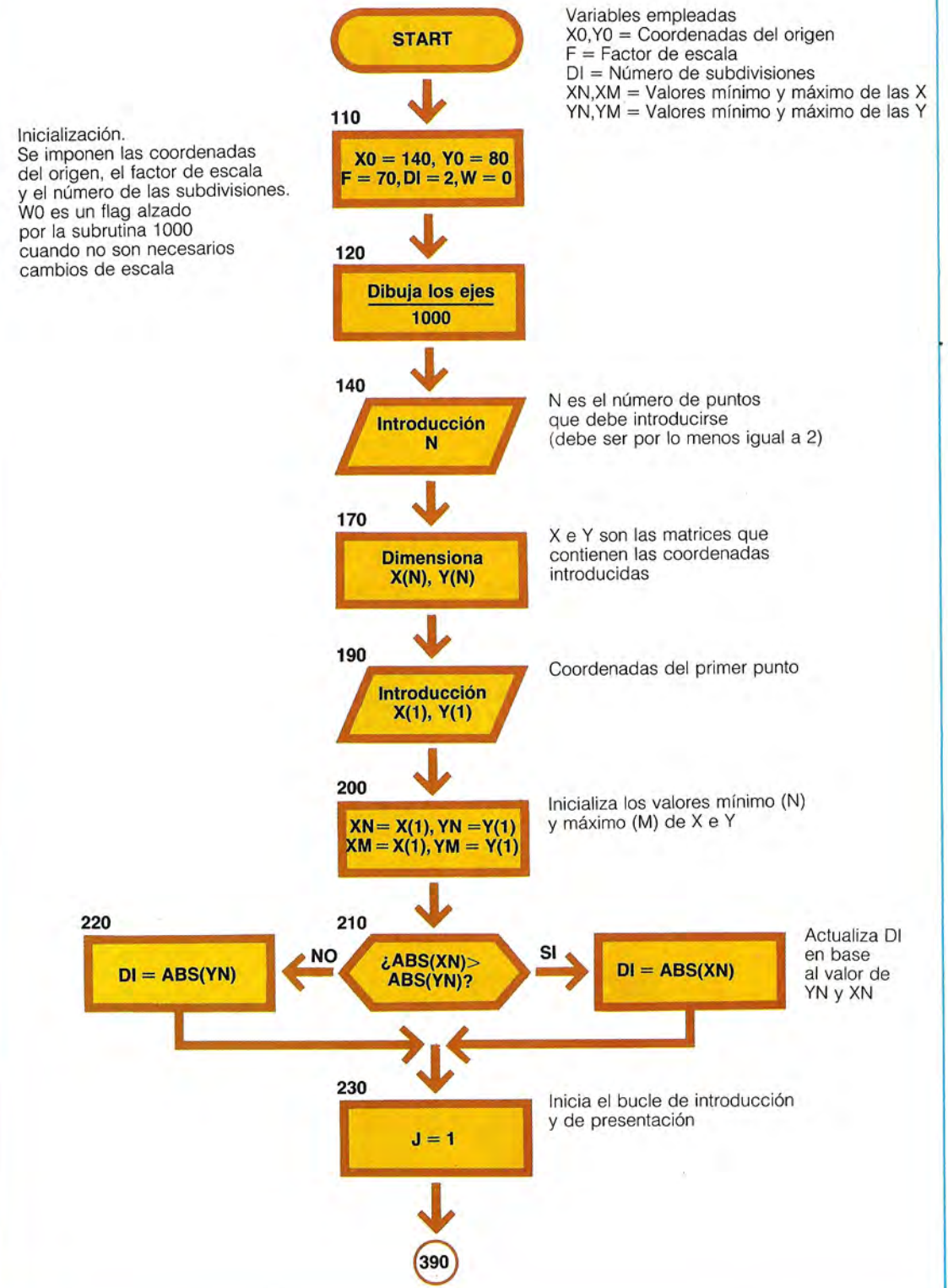
CALCULO DE LOS COEFICIENTES A Y B DE LA RECTA DE REGRESION

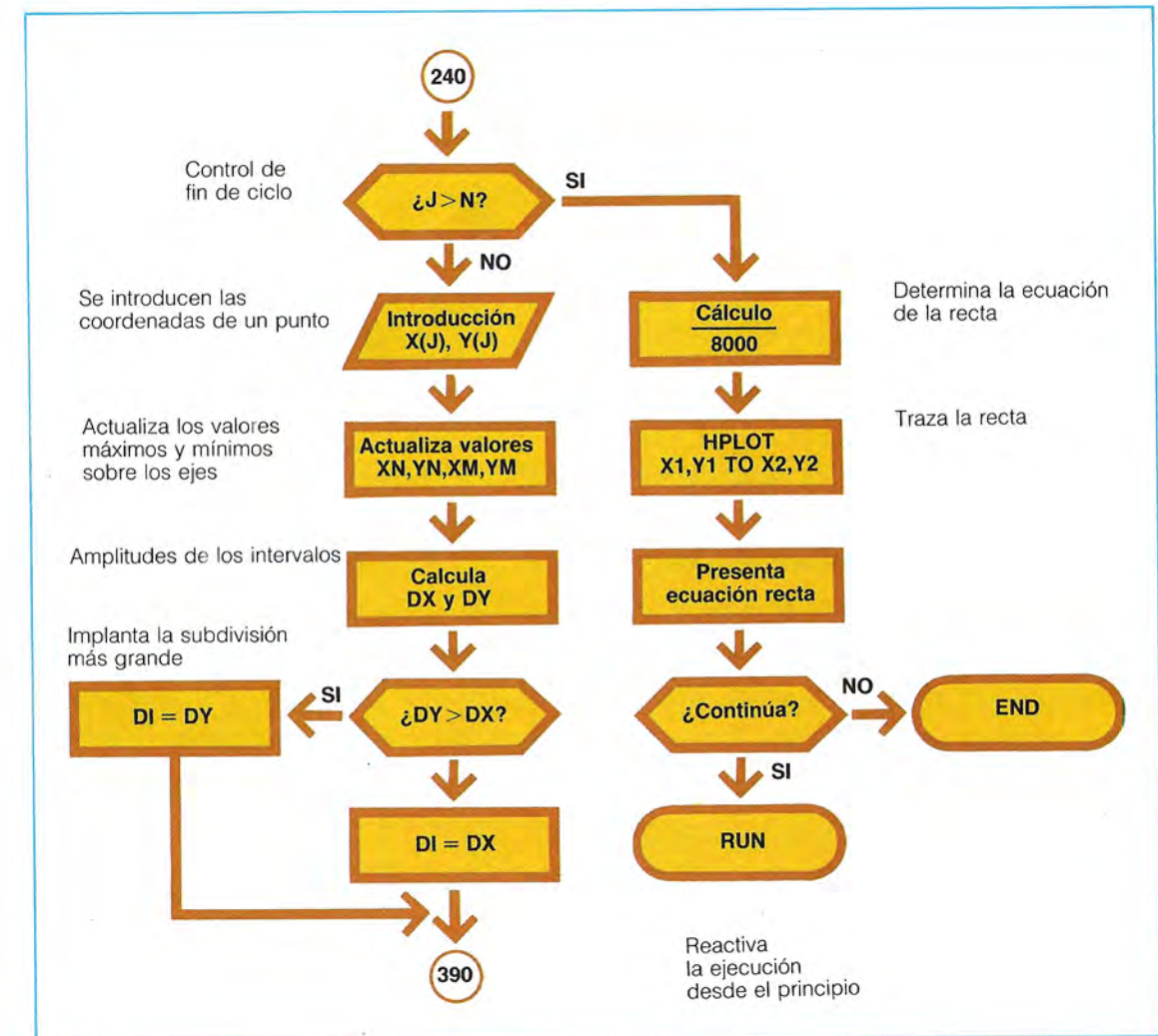
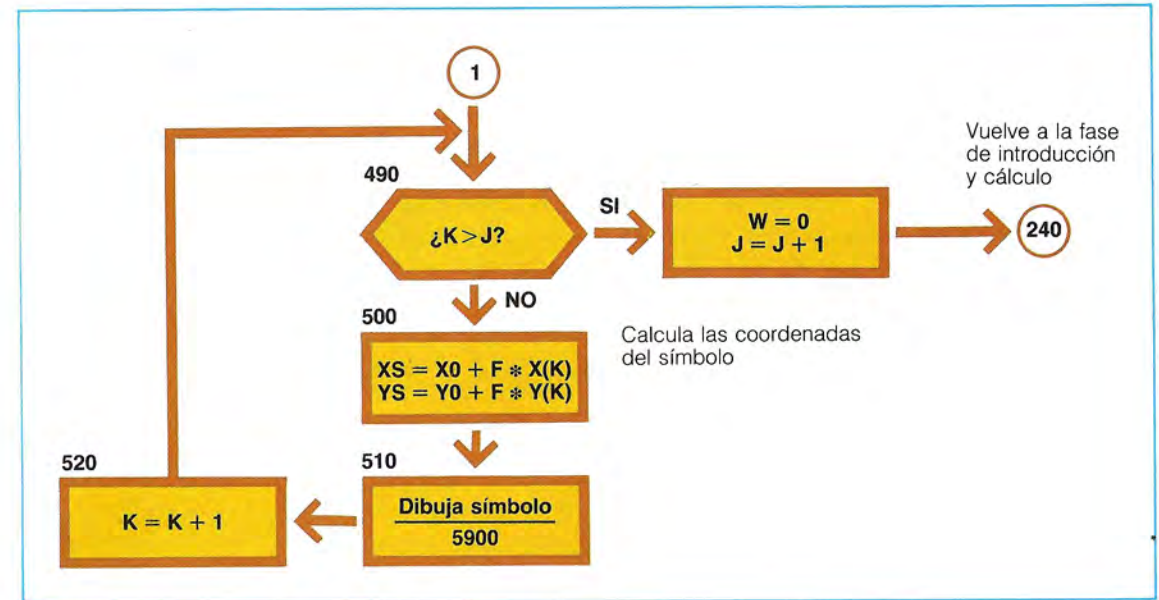
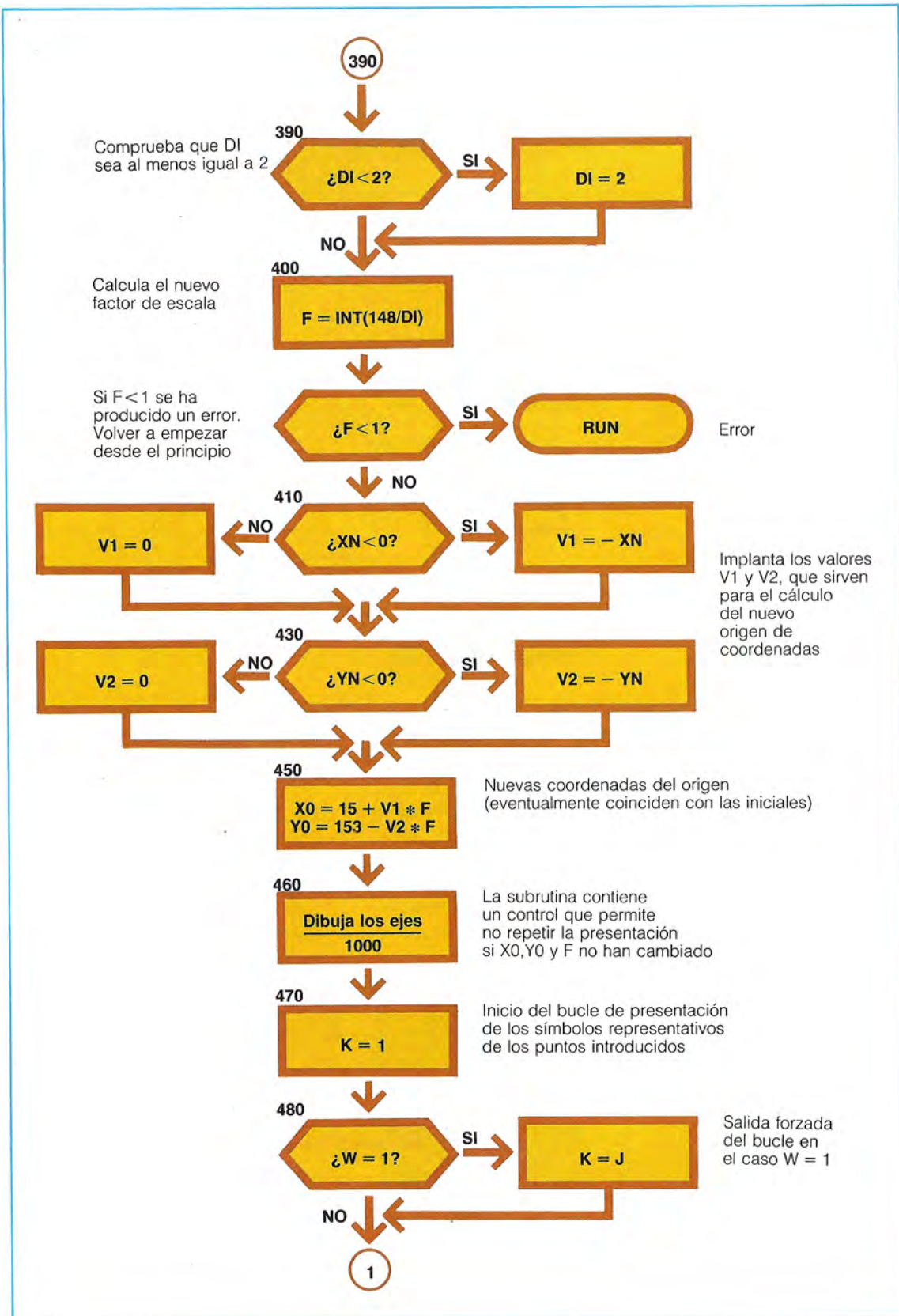


REGRESION LINEAL DE MINIMOS CUADRADOS



RECTA DE REGRESION





gen. La subrutina 1000 contiene un control (flag W) que permite no repetir la presentación de los ejes en el caso en que el factor de escala y las coordenadas del origen no varíen con respecto al paso anterior.

La última función es la que vuelve a dibujar todos los puntos, siempre bajo la condición de que se haya variado el factor de escala. Obsérvese particularmente la salida forzada del bucle. Si el factor de escala varía, es necesario un bucle que represente todos los puntos anteriores; este bucle debe activarse si la escala no ha variado, o sea si el flag W vale 1. La función se obtiene poniendo, bajo condición, el índice del

bucle en el extremo superior (IF W = 1 THEN K = J). Sin embargo, este método tiene una posibilidad de aplicación limitada a una categoría particular de máquinas, y en el caso más general no puede utilizarse. La modificación que permite su generalización consiste simplemente en la introducción de un by-pass en todo el bucle bajo la condición W = 1. El listado de todo el programa en la versión en el Personal Kid (Simpel 2010, Apple y compatibles) puede verse en esta página y en las siguientes. Para su transferencia a otras máquinas deben modificarse los límites de la pantalla y sustituirse algunas instrucciones, como HOME, HGR, etc.

TRAZADO DE LA RECTA DE REGRESION

```

10  REM -----
20  REM RECTAS DE REGRESION
30  REM -----
40  REM
50  REM
60  REM
70  FOR J = 1 TO 4
80  READ SX(J), SY(J)
90  NEXT J
100 DATA -1,1,1,1,1,-1,-1,-1
110 XO = 140
    YO = 80
    F = 70
    DI = 2
    W = 0
120 GOSUB 10000
130 HOME
    VTAB 22
140 INPUT "NUMERO DE PUNTOS?";N
150 N = INT (N)
160 IF N < 2 THEN 130
170 DIM X(N), Y(N)
180 HOME
    VTAB 22
190 INPUT "PUNTO N.1 ";X(1),Y(1)
200 XN = X(1)
    XM = X(1)
    YN = Y(1)
    YM = Y(1)
210 IF ABS (XN) > ABS (YN) THEN DI
    = ABS (XN)
    GOTO 230
220 DI = ABS (YN)
230 J = 1
    GOTO 390
240 IF J > N THEN 540
250 HOME
    VTAB 22
260 PRINT "PUNTO N. "J": ";
    INPUT " ";X(J),Y(J)
270 IF X(J) > XM THEN XM = X(J)
280 IF X(J) < XM THEN XM = X(J)
290 IF Y(J) > YM THEN YM = Y(J)
300 IF Y(J) < YM THEN YM = Y(J)
310 IF XM > 0 AND XN < 0 THEN DX = XM
    - XN
    GOTO 340

```

```

320 IF XN >= 0 AND XM > 0 THEN DX
    = XM
    GOTO 340
330 DX = - XN
340 IF YN < 0 AND YM > 0 THEN DY = YM
    - YN
    GOTO 370
350 IF YN >= 0 AND YM > 0 THEN DY
    = YM
    GOTO 370
360 DY = -YN
370 IF DY > DX THEN DI = DY
    GOTO 390
380 DI = DX
390 IF DI < 2 THEN DI = 2
400 F = INT (148 / DI)
    IF F < 1 THEN RUN
410 IF XN < 0 THEN V1 = - XN
    GOTO 430
420 V1 = 0
430 IF YN < 0 THEN V2 = - YN
    GOTO 450
440 V2 = 0
450 XO = V1 * F + 15
    YO = 153 - V2 * F
460 GOSUB 1000
470 K = 1
480 IF W = 1 THEN K = J
490 IF K > J THEN W = 0
    GOTO 530
500 XS = X(K) * F + XO
    YS = YO - X(K) * F
510 GOSUB 5900
    REM SIMBOLO
520 K = K + 1
    GOTO 490
530 J = J + 1
    GOTO 240
540 GOSUB 8000
    REM CALCULO
550 HPLOT X1,Y1 TO X2,Y2
560 S$ = "-"
570 IF A > 0 THEN S$ = "+"
580 HOME
    VTAB 22
590 PRINT "LA RECTA APROXIMADA ES:
    "
600 IF A = 0 AND B = 0 OR D = 0
    THEN INVERSE
    PRINT "X = ":XN
    GOTO 630
610 IF B = 0 THEN INVERSE
    PRINT "Y = "A
    GOTO 630
620 INVERSE
    PRINT "Y = "B" * X "S$" " ABS (A)
630 NORMAL
640 VTAB 24
650 PRINT "QUIERE CONTINUAR? (S/N) ";
    GET A$
660 IF A$ = "S" THEN RUN
670 TEXT
    HOME

```

```

END
997 REM -----
998 REM DIBUJA EJES
999 REM -----
1000 IF XO = XP AND YO = YP AND F = FF
      THEN W = 1
      RETURN
1010 HGR
      HCOLOR= 3
1020 HPLOT 0,0 TO 279,0 TO 279,159
      TO 0,159 TO 0,0
1030 HPLOT 15,YO TO 265,YO
      HPLOT XO,153 TO XO,5
1040 N2 = 5
      FOR N1 = 2 TO 0 STEP - 1
      HPLOT XO - N1,N2 TO XO + N1,N2
      N2 = N2 - 1
      NEXT N1
1050 N2 = 265
      FOR N1 = 2 TO 0 STEP - 1
      HPLOT N2,YO - N1 TO N2,YO + N1
      N2 = N2 + 1
      NEXT N1
1060 HPLOT 271,YO - 2 TO 276,YO + 3
      HPLOT 276,YO - 2 TO 271,YO + 3
1070 HPLOT XO - 10,3 TO XO - 7,6
      HPLOT XO - 4,3 TO XO - 10,8
1080 IF F = 1 THEN 1220
1090 FOR G = XO TO 15 STEP - F
1100 HPLOT G,YO + 1 TO G,YO - 1
1110 NEXT G
1120 FOR G = XO TO 265 STEP F
1130 HPLOT G,YO+1 TO G,YO-1
1140 NEXT G
1150 FOR G = YO TO 5 STEP - F
1160 HPLOT XO + 1,G TO XO - 1,G
1170 NEXT G
1180 FOR G = YO TO 153 STEP F
1190 HPLOT XO ' 1+G TO XO - 1,G
1200 NEXT G
1210 YP = XO
      YF = XO
      FF = F
1220 RETURN
5899 REM -----
5900 REM SIMBOLO
5901 REM
5910 S = F / 4
      IF S > 5 THEN S = 5
5920 IF S < 2 THEN S = 2
5930 XA = SX(1) * S + XS
      XA = SY(1) * S + YS
5940 XC = XA
      YC = YA
5950 FOR T = 2 TO 4
5960 XB = SX(T) * S + XS
      YB = SY(T) * S + YS
5970 HPLOT XA,YA TO XB,YB
5980 XA = XB
      YA = YB
5990 NEXT T
6000 HPLOT XB,YB TO XC,YC
6010 RETURN

```

```

7999 REM -----
8000 REM CALCULO
8010 REM -----
8020 C1 = 0
      C2 = 0
      C3 = 0
      C4 = 0
8030 FOR I = 1 TO N
8040 C1 = C1 + X(I)
8050 C2 = C2 + Y(I)
8060 C3 = C3 + X(I) ^ 2
8070 C4 = C4 + X(I) * Y(I)
8080 NEXT I
8090 D = N * C3 - C1 ^ 2
8100 IF D = 0 THEN Y1 = YO - YN * F
      : Y2 = YO - YM * F
      : X1 = XO + XN * F
      : X2 = X1
      : RETURN
8110 A = (C2 * C3 - C1 * C4) / D
8120 B = (N * C4 - C1 * C2) / D
8130 IF B = 0 THEN Y1 = YO - A * F
      : Y2 = Y1
      : X1 = XO + XN * F
      : X2 = XO + XM * F
      : RETURN
8140 IF A = 0 AND B = 0 THEN Y1 = YO
      : -YN * F
      : Y2 = YO - YN * F
      : X1 = XO + XN * F
      : X2 = X1
      : RETURN
8150 WS = 0
8160 X = (YM - A) / B
      Y = YM
8170 IF X >= XN AND X <= XM THEN
      GOSUB 8270
8180 X = (YN - A) / B
      Y = YN
8190 IF X >= XN AND X <= XM THEN
      GOSUB 8270
8200 X = XM
      Y = B * XM + A
8210 IF Y >= YN AND Y <= YM THEN
      GOSUB 8270
8220 X = XN
      Y = B * XN + A
8230 IF Y >= YN AND Y <= YM THEN
      GOSUB 8270
8240 X1 = XO + X1 * F
      X2 = XO + X2 * F
8250 Y1 = YO - Y1 * F
      Y2 = YO - Y2 * F
8260 RETURN
8270 IF WS = 1 THEN 8290
8280 X1 = X
      : Y1 = Y
      : WS = 1
      : RETURN
8290 X2 = X
      : Y2 = Y
      : RETURN

```

REGRESION LINEAL DE MINIMOS CUADRADOS

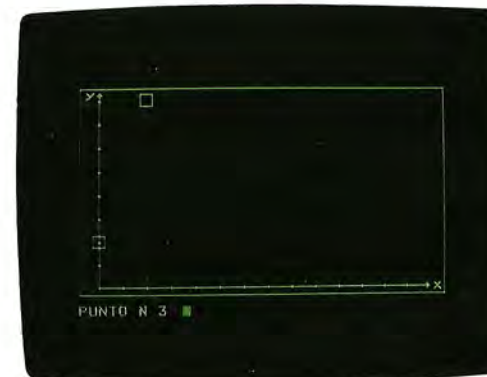
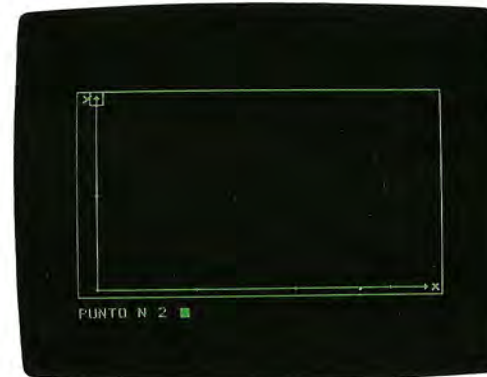
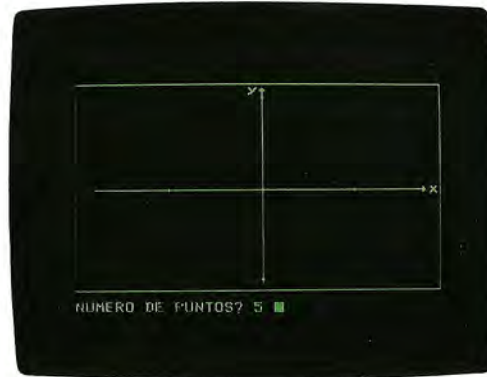
El sistema presenta los ejes cartesianos y pide el número de puntos.
Para comprobar la precisión del software, se introducen cinco puntos que pertenecen a la recta $Y = 3 * X + 2$.

	X	Y
Punto 1	0	2
Punto 2	2	8
Punto 3	3	11
Punto 4	5	17
Punto 5	10	32

En el sistema ya se ha introducido el primer punto de coordenadas 0,2. El programa adapta la posición de los ejes cartesianos de manera que se disponga, para la representación de los valores introducidos, de toda el área de la pantalla. Respecto a las fotos anteriores, el origen se ha desplazado hacia abajo y hacia la izquierda, puesto que los valores introducidos (por ahora sólo el punto 0,2) no contienen números negativos; la pantalla por tanto queda completamente dedicada al cuadrante positivo del sistema de referencia, y el valor máximo de Y es igual al valor máximo introducido.

Después de la introducción del punto 2 (con coordenadas 2,8), el programa reduce la escala del eje Y de manera que se adecue a la nueva abscisa (Y = 8). El punto anterior parece ahora situado abajo respondiendo a las mismas coordenadas. Efectivamente, en la foto anterior, el eje Y estaba dividido sólo en dos partes (trazos horizontales) y el símbolo ocupaba la posición 2 (o sea la ordenada Y = 2), mientras que en la nueva situación, el programa ha modificado automáticamente la escala calibrándola sobre el nuevo valor máximo introducido (Y = 8).

Al final de las introducciones (cuyo número se ha declarado al principio), el programa presenta el gráfico final, que comprende tanto los puntos introducidos como la recta de regresión; además se visualiza la expresión analítica de la recta. Respecto a los valores utilizados para determinar los puntos, los coeficientes calculados no presentan ninguna diferencia. El valor 1, que aparece al final de una serie de ceros después de la coma, se debe a la ausencia de truncados en la presentación de los coeficientes. En las aplicaciones prácticas puede truncarse el resultado a la segunda o a la tercera cifra decimal, y en este caso no se tiene ninguna diferencia.



El segundo ejemplo, al contrario del primero, se refiere a un cálculo de regresión propiamente dicho, realizado sobre los siguientes valores:

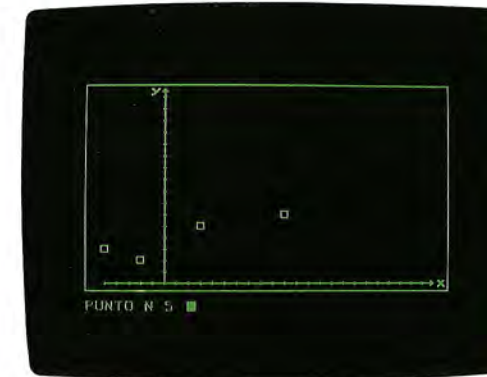
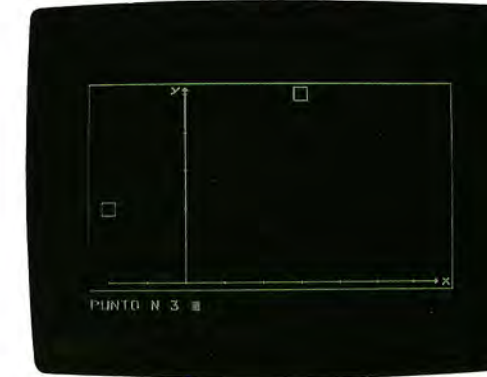
	X	Y
Punto 1	3	5
Punto 2	-2	6
Punto 3	10	6
Punto 4	-5	3
Punto 5	1	-2

Se han introducido las coordenadas del primer punto, y el programa queda a la espera del segundo dato.

La introducción del primer valor negativo (punto 2, coordenadas -2,2) produce el desplazamiento del eje Y hacia la derecha, siendo necesaria una porción del eje X negativa (-2).

Las sucesivas introducciones determinan la adecuación del campo presentado (posición de origen y factor de escala). En la foto se ha introducido el cuarto punto.

La introducción del último punto (1, -2) activa el cálculo y la presentación de la recta de regresión. Como puede observarse, en este caso, los valores resultan muy dispersos, y la recta de regresión proporciona una aproximación relativamente precisa.



Programa para el trazado del gráfico de una función cualquiera

La representación de una función general necesita muchos más dispositivos que los que deben adoptarse en el caso de la representación de una recta. Las principales implantaciones que deben preverse son las siguientes:

- cálculo del factor de escala y su normalización
- control y recuperación de los errores
- simbología a elección del usuario

El factor de escala puede calcularse muy sencillamente como cociente entre las dimensiones de la pantalla (según los dos ejes) y los correspondientes intervalos de los valores a representar. Por ejemplo, para representar una función que asume valores comprendidos entre 20 y 70 sobre una pantalla que tenga 200 puntos según el eje Y, el factor de escala es $200/(70 - 20) = 200/50 = 4$. De este valor puede calcularse la ordenada Y0 del origen, que debe coincidir con el valor mínimo de la Y(20) más un margen que sirve para evitar que los ejes se dibujen demasiado cerca del borde de la pantalla. Para la

abscisa X puede procederse análogamente. En general, el resultado no será un número entero; en el ejemplo anterior basta que los valores asumidos por la función varíen entre 20 y 71 para dar un factor de escala con decimales y, por tanto, de difícil interpretación.

Para evitar este inconveniente debe preverse una rutina de normalización, o sea la elección del valor más importante entre algunos definidos previamente por el usuario. Por ejemplo, en el caso anterior de función entre los valores 20 y 71, el factor de escala $200/51 = 3.9$ debe normalizarse a 3. Así se tiene un gráfico de dimensiones más reducidas (no se aprovecha toda la pantalla), pero de fácil interpretación.

La función puede asumir valores tanto negativos como positivos, y por tanto debe adecuarse la posición del origen (X0,Y0) para permitir el máximo aprovechamiento de la pantalla. La función puede tener además una forma que proporcione valores muy pequeños o muy grandes, que superen el límite de la presentación. Este problema puede resolverse introduciendo un factor de multiplicación igual a una potencia de 10. Por ejemplo, si se debe representar una función que varía entre 0.03 y 0.07, antes de represen-

Presentación de rectángulos superpuestos en un monitor en colores.



Centro THC/Marka

PRINCIPALES VARIABLES UTILIZADAS EN EL PROGRAMA

V1	= Máximo factor de escala previsto
SX(4), SY(4)	= Desplazamientos para trazar el símbolo
VN/10)	= Factores de escala previstos (para normalización)
K\$	= Flag que indica eje X o eje Y
XS,YS	= Coordenadas expresadas en puntos de pantalla
XM,XN	= Valores máximo y mínimo eje X
YM,YN	= Valores máximo y mínimo de la función (Y)
FX,FY	= Factores de escala eje X y eje Y
DX	= Paso eje X
NP	= Número de puntos a presentar
LX,LY	= Longitud de los ejes
ER	= Código del último error comprobándose
LOC	= Número de la línea en que se ha producido el error
K	= Potencia de 10 en el factor de escala
S1,S2	= Factor de escala para los símbolos
T\$	= Tipo de gráfico (1 para puntos, 2 para segmentos, 3 para símbolos)
X0,Y0	= Coordenadas del origen de los ejes

tarla puede referirse a los valores 3 y 7 introduciendo un valor de multiplicación $K = 10^2$. Naturalmente, en el gráfico debe indicarse el valor de K utilizado, o aún mejor, el valor por el que deben multiplicarse las coordenadas.

El control y la recuperación de los errores son necesarios en los casos en que la función asume valores indeterminados o incluso no representables.

Por ejemplo, la función:

$$Y = \frac{1}{X - 3}$$

asume valor infinito en el punto $X = 3$. El programa debe poder detectar esta anomalía y, por tanto, evitar el valor de la abscisa (3) que la origina; sin estos controles se tendría un error de desbordamiento.

La simbología definida por el usuario es una implantación no indispensable, pero a veces útil a los fines de una presentación gráfica más agradable. En el programa que presentamos en estas páginas se han previsto tres sistemas de presentación:

- por puntos
- por segmentos
- por símbolos

En el primer caso, cada punto de la función está representado por un punto de pantalla, y el gráfico se presenta como una sucesión de puntos más o menos cercanos. En la segunda forma,

cada punto está unido al que le precede y al que le sigue mediante dos segmentos, y el proceso de la función se presenta entonces como una línea quebrada. El último tipo de representación se obtiene trazando un cuadrado en el punto de pantalla correspondiente a las coordenadas de cada punto de la curva.

En la página siguiente se ha representado el diagrama de flujo del programa.

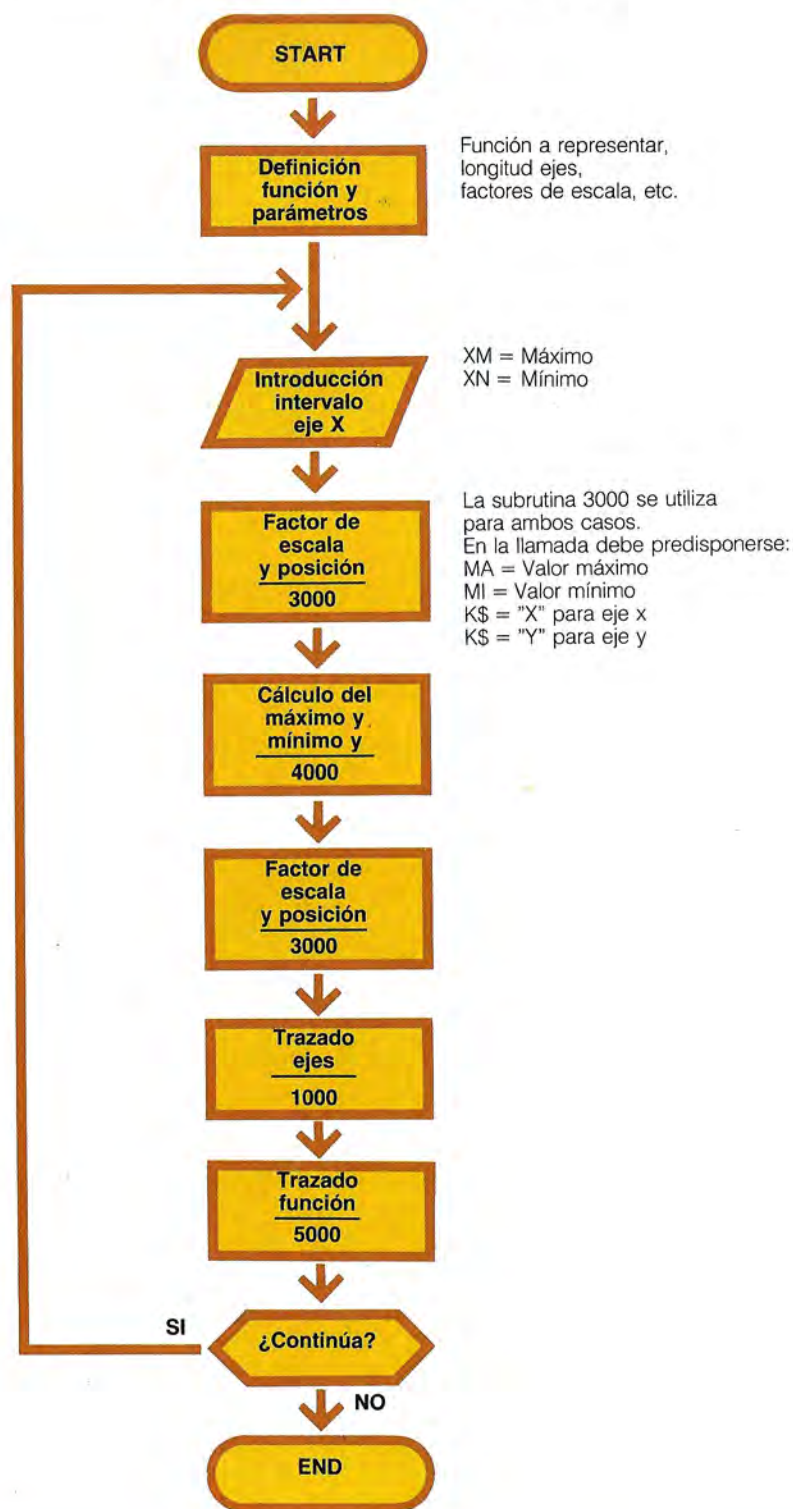
El primer bloque define los parámetros característicos de la máquina (dimensiones de la pantalla, factores de escala, etc.) y la función a representar. La variación de la función se produce con una conversación que permite la sustitución de la línea que contiene la definición (línea 100 DEF FN...; este procedimiento puede utilizarse sólo en Basic interpretado). A continuación, el programa pide el intervalo XM, XN de los valores asumidos por la variable independiente, y para estos valores calcula el factor de escala en el eje X (subrutina 3000).

En el campo XN a XM se realiza después la búsqueda de los valores máximo y mínimo de la variable dependiente (subrutina 4000), y el cálculo del correspondiente factor de escala.

Las últimas dos funciones corresponden al trazado de los ejes (1000) y del gráfico (5000) con la simbología seleccionada.

En la tabla de arriba se han representado las principales variables utilizadas, y en las páginas que siguen, el diagrama de flujo de las subrutinas insertadas en el programa, algunas de las cuales se describirán más adelante.

DIAGRAMA DE FLUJO DEL PROGRAMA DE TRAZADO DEL GRAFICO DE UNA FUNCION



Cálculo y normalización del factor de escala.

La subrutina de cálculo del factor de escala (ver la figura de la página siguiente) utiliza el flag K\$ para seleccionar el eje (X o Y) sobre el que se quiere realizar el proceso. En la salida proporciona el factor de escala para cada eje (FX,FY), las coordenadas X0,Y0 del origen y el factor multiplicador K.

En la rutina se ha previsto el redondeo de los valores máximo y mínimo utilizando números enteros.

La normalización se realiza con una llamada a la subrutina 6000 (ver la figura de la pág. 1467).

Búsqueda de los valores máximo y mínimo de Y (subrutina 4000).

El diagrama de flujo de la subrutina puede verse en la pág. 1468. El proceso se realiza calculando el valor de Y en varios puntos del intervalo y memorizando cada vez el valor más elevado y el más bajo.

Presentación del proceso (subrutinas 5000, 7000).

Puede realizarse en uno de los tres modos previstos (puntos, segmentos, símbolos). La presentación por puntos la realiza la subrutina 7000, mientras que la por segmentos o por

símbolos la realiza la 5000 (el diagrama de flujo representado en la figura de la pág. 1469 sólo se refiere a esta última opción).

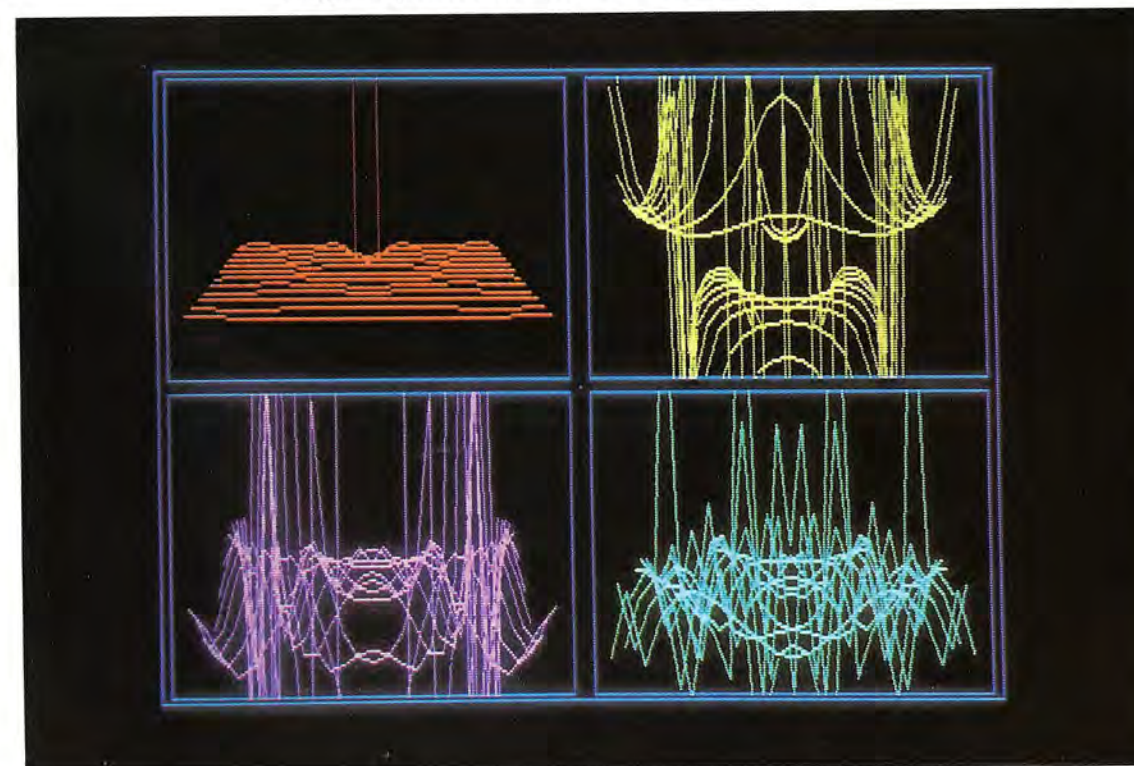
La subrutina 7000 es análoga, con exclusión de la llamada 5900, que sirve para trazar el símbolo (ver la figura de la pág. 1470).

Gestión de los errores (subrutina 10000).

En las figuras de las pág. 1471 y 1472 se ha representado el diagrama de flujo de la subrutina de gestión de los errores. Ésta es la única parte estrechamente ligada a la máquina y que difícilmente puede generalizarse. La lógica y el listado se refieren al Personal Kid (Siprel 2010, Apple y compatibles); para otras máquinas, por lo menos deben modificarse los códigos de error y las posiciones de memoria.

Al producirse un error, en el sistema utilizado, se escribe en una determinada posición de memoria (222) el código numérico correspondiente a aquel tipo de error y, en otra posición, el número de la línea en que se ha generado el error. Indicando con ER la variable que deberá contener el código de error y con LOC la variable que contendrá el número de la línea, las instrucciones para obtener estos valores son

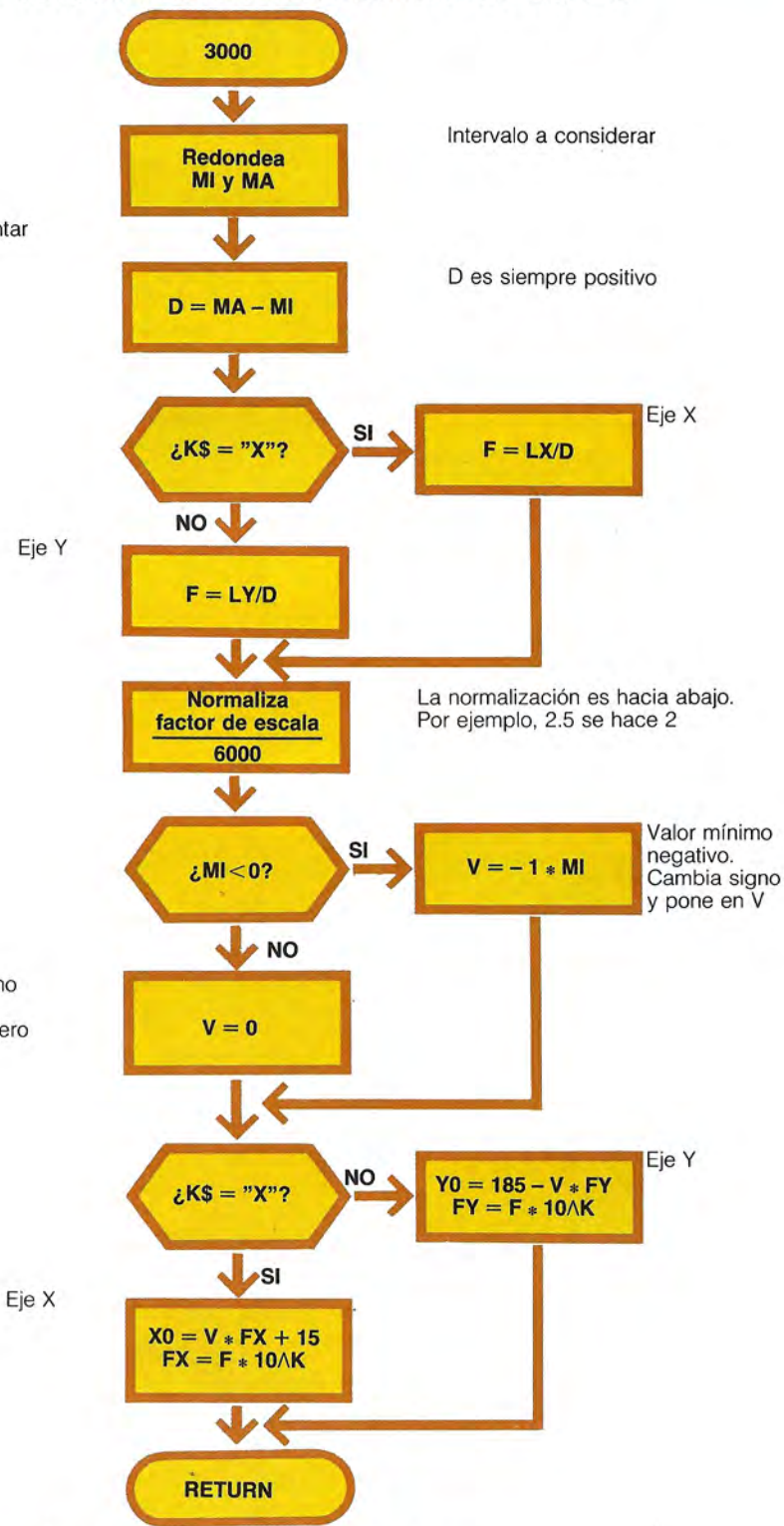
Imágenes gráficas tridimensionales en colores con uso de las ventanas vídeo.



CALCULO Y NORMALIZACION DEL FACTOR DE ESCALA

Entradas:
 MA = Valor máximo a representar
 MI = Valor mínimo
 K\$ = "X" para el eje X
 "Y" para el eje Y

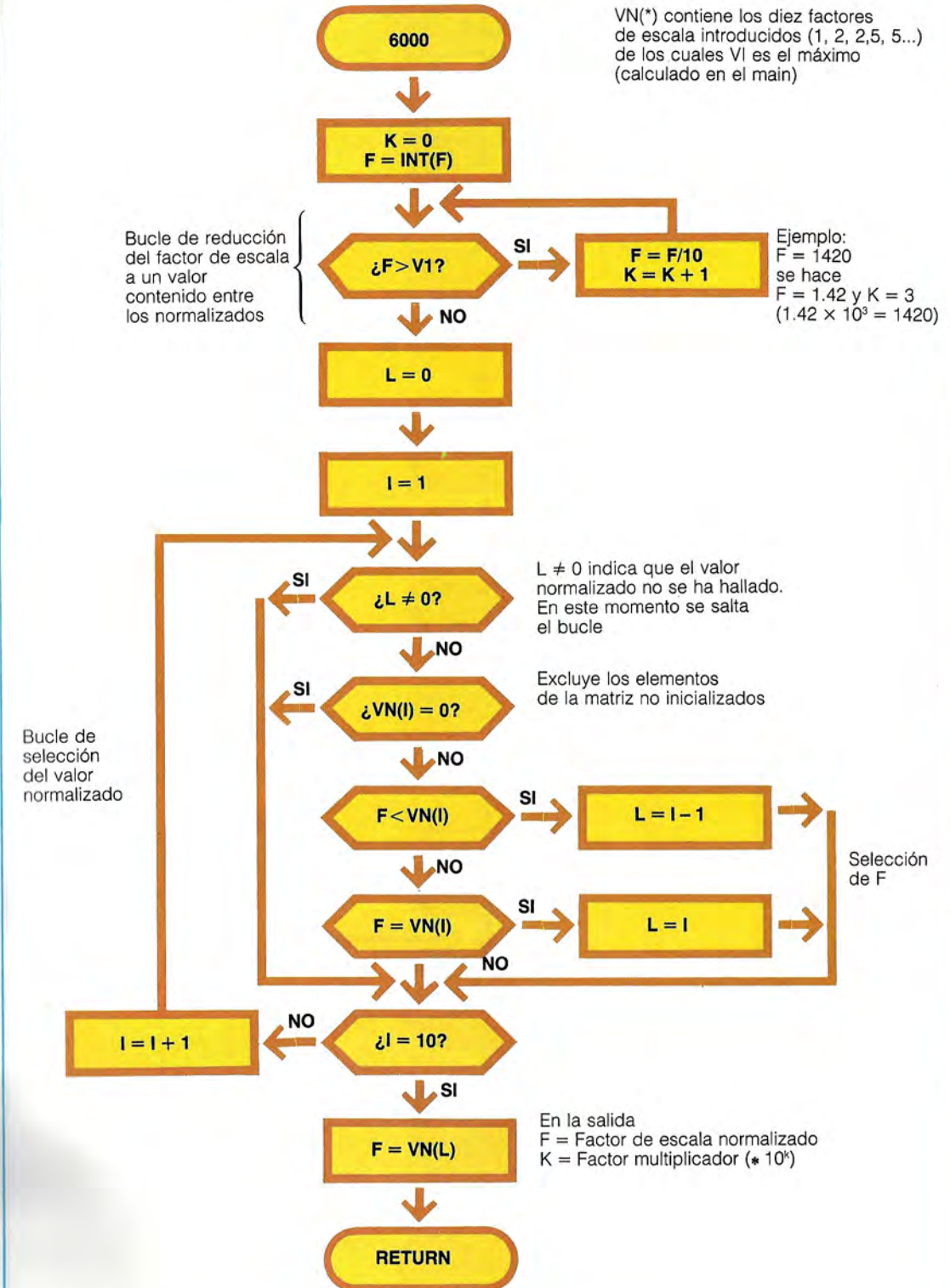
Salidas:
 F = Factor de escala
 X0, Y0 = Coordenadas del origen



La subrutina se utiliza dos veces, una para cada eje. La selección se obtiene con el flag K\$

NORMALIZACION DEL FACTOR DE ESCALA

VN(*) contiene los diez factores de escala introducidos (1, 2, 2,5, 5...) de los cuales V1 es el máximo (calculado en el main)



Bucle de reducción del factor de escala a un valor contenido entre los normalizados

Ejemplo:
 F = 1420
 se hace
 F = 1.42 y K = 3
 (1.42 × 10³ = 1420)

L ≠ 0 indica que el valor normalizado no se ha hallado. En este momento se salta el bucle

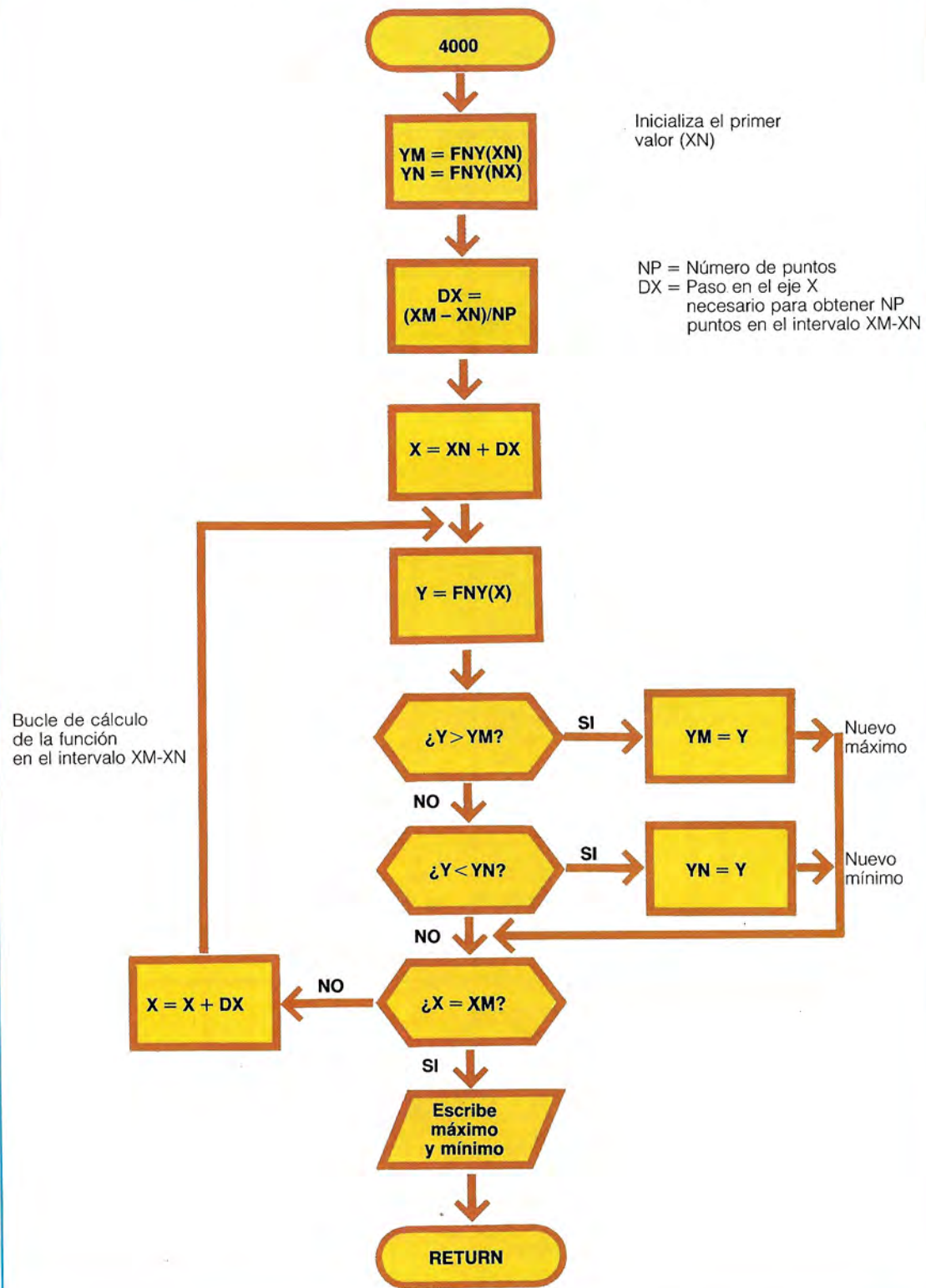
Excluye los elementos de la matriz no inicializados

Bucle de selección del valor normalizado

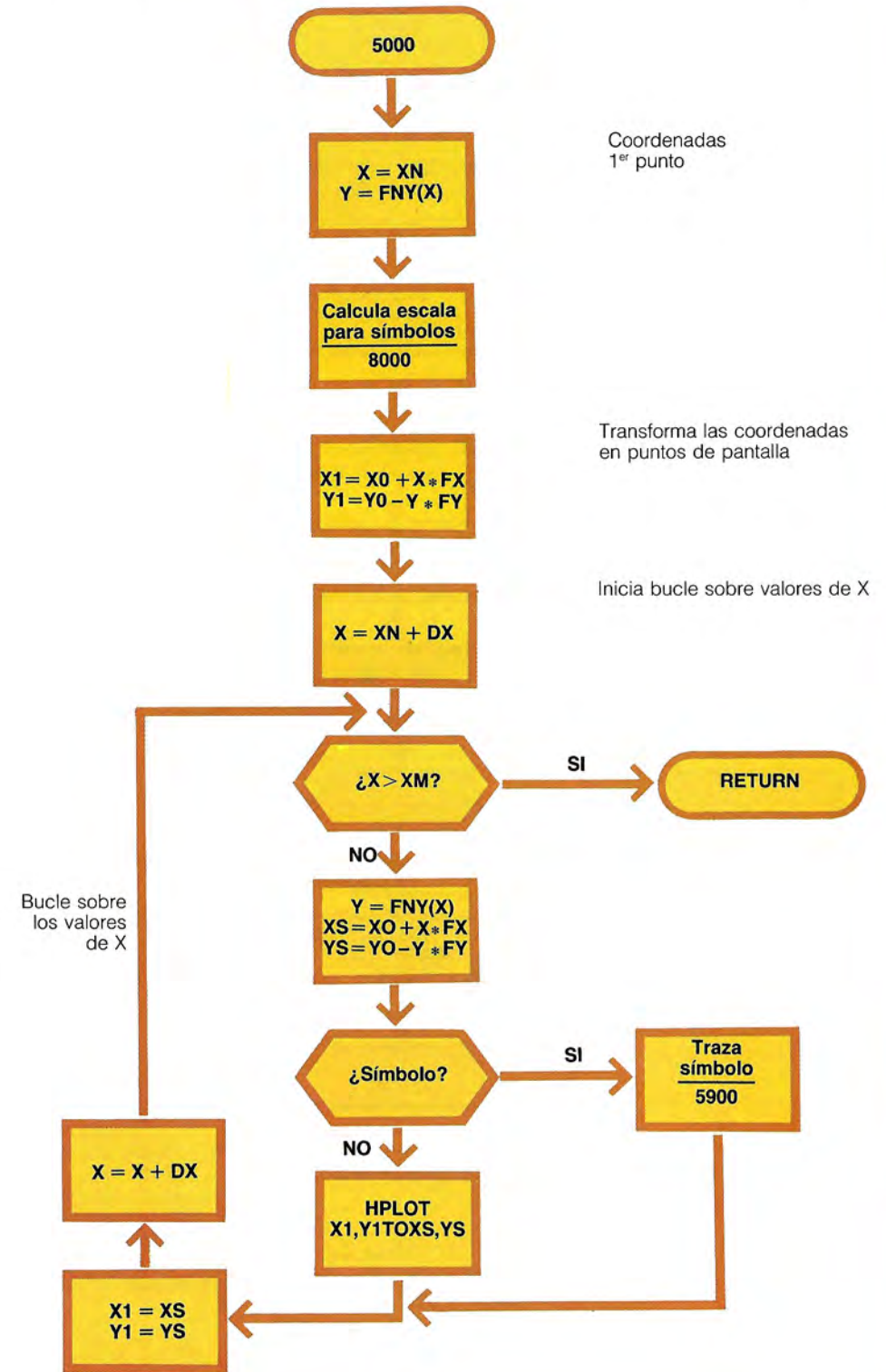
Selección de F

En la salida
 F = Factor de escala normalizado
 K = Factor multiplicador (* 10^K)

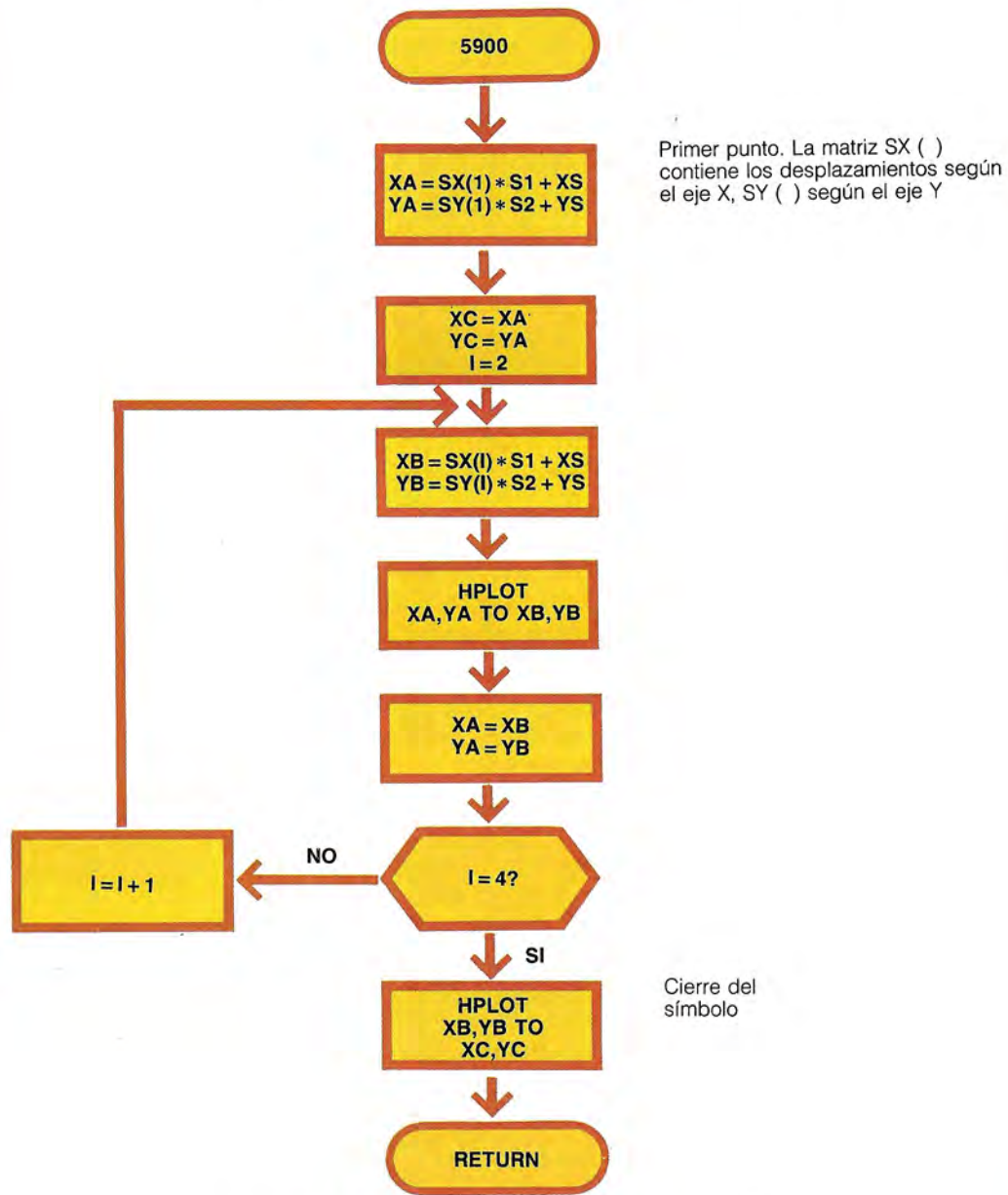
BUSQUEDA DEL MAXIMO Y DEL MINIMO



PRESENTACION DE LA CURVA POR SEGMENTOS O POR SIMBOLOS

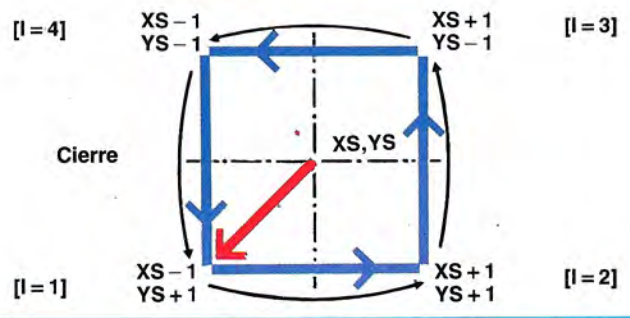


TRAZADO DE UN SIMBOLO CON CENTRO EN XS, YS

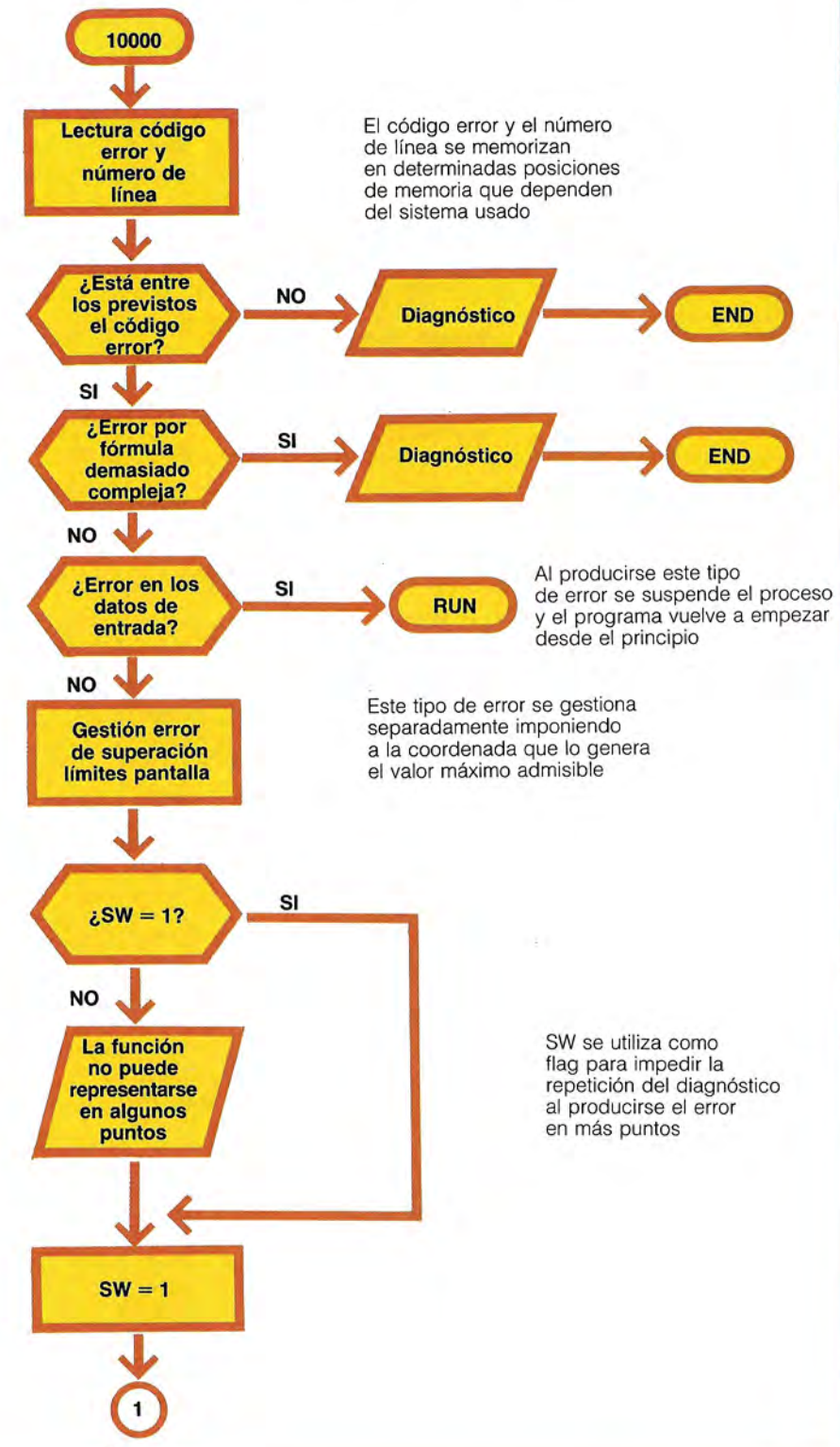


Primer punto. La matriz SX () contiene los desplazamientos según el eje X, SY () según el eje Y

Cierre del símbolo



RUTINA DE GESTION DE LOS ERRORES

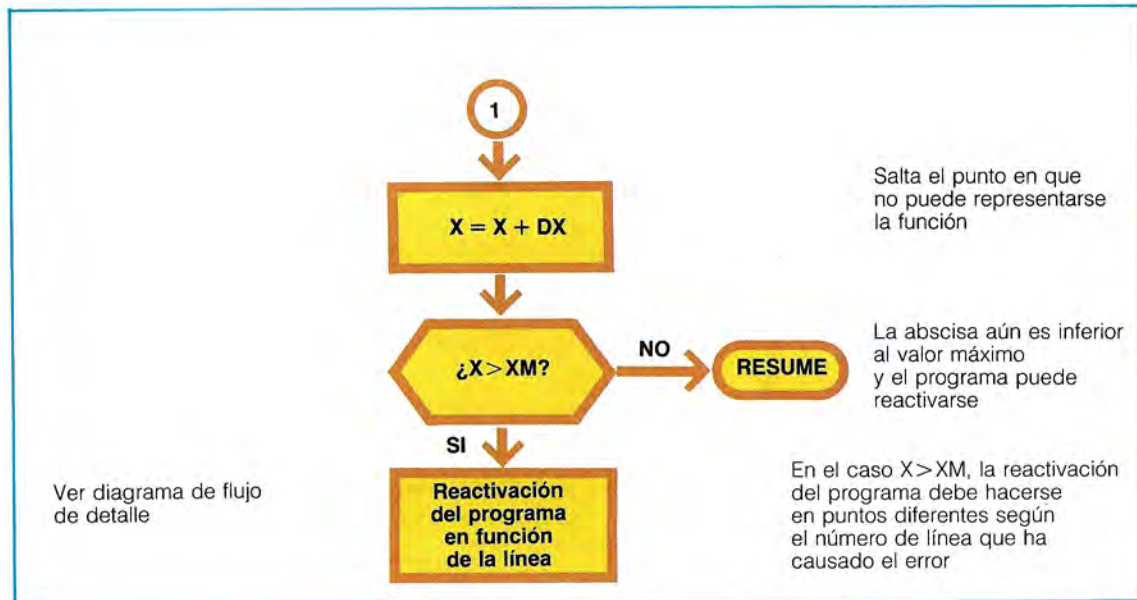


El código error y el número de línea se memorizan en determinadas posiciones de memoria que dependen del sistema usado

Al producirse este tipo de error se suspende el proceso y el programa vuelve a empezar desde el principio

Este tipo de error se gestiona separadamente imponiendo a la coordenada que lo genera el valor máximo admisible

SW se utiliza como flag para impedir la repetición del diagnóstico al producirse el error en más puntos



Ver diagrama de flujo de detalle



ER = PEEK (222)
 LOC = PEEK (218) + PEEK (219) * 256

La primera tiene un significado inmediato: la instrucción PEED toma el contenido de la memoria 222 y lo transfiere a la variable ER. En la segunda instrucción, el dato (dirección de la línea) debe tomarse de dos memorias sucesivas (218 y 219) y reconstruirse. Cada memoria contiene 8 bits, mientras que el direccionamiento al número de la línea necesita 16; por tanto, el sistema divide este número en dos partes, que coloca en dos memorias sucesivas. El producto por 256 es necesario para atribuir a la primera parte de la dirección el peso adecuado con respecto a la segunda (desplazar 8 bits equivale a

multiplicar por 256). Los códigos de error reconocidos, válidos sólo para la máquina utilizada, son los siguientes:

- 53 = Cantidad ilegal
- 133 = División por cero
- 254 = Error de introducción
- 69 = Desbordamiento
- 191 = Fórmula demasiado compleja

Para otras máquinas, si existe una gestión de los errores similar a la descrita, deberá comprobarse la correspondencia de los códigos, modificándolos si es preciso. El listado del programa se ha representado en las páginas 1473 a 1478, y en las páginas 1479 a 1481 pueden verse algunos ejemplos de funcionamiento.

TRAZADO DEL GRAFICO DE UNA FUNCION

```

90 REM -----
91 REM FUNCIONES MATEMATICAS
92 REM -----
93 REM
94 REM
95 REM
96 REM
97 REM -----
98 REM INICIALIZACION
99 REM -----
100 DEF FN Y(X) = X * SEN (X)
110 LX = 250
    : LY = 180
    : SW = 0
120 FOR J = 1 TO 4
    : READ SX(J),SY(J)
    : NEXT J
130 DATA -1,1,1,1,1,-1,-1,-1
140 ONERR GOTO 10000
150 V1 = 1
160 FOR J = 1 TO 10
    : READ VN(J)
    : IF VN(J) > V1 THEN V1 = VN(J)
170 : NEXT J
180 DATA 1,2,3,4,5,6,7,8,9,0
190 REM -----
200 REM MENU
210 REM -----
220 TEXT
    : HOME
    : GOSUB 2000
230 : HTAB 3
    : VTAB 5
    : INPUT "INTERVALO EJE X? ";XN,XM
    : HTAB 39
    : VTAB 5
    : PRINT "*"
240 : IF XN <= XM THEN 220
250 : IF XM - XN > 250 THEN 220
260 : HTAB 3
    : VTAB 10
    : INPUT "NUMERO DE PUNTOS? ";NF
    : HTAB 39
    : VTAB 10
    : PRINT "*"
  
```

```

270 HTAB 3
: VTAB 15
: PRINT " ELIJA EL TIPO DE GRAFICO:
"
280 HTAB 3
: PRINT "1) POR PUNTOS"
290 HTAB 3
: PRINT "2) POR SEGMENTOS"
300 HTAB 3
: PRINT "3) POR SIMBOLOS"
310 VTAB 15
: HTAB 30
: INPUT " ";T$
: VTAB 15
: HTAB 39
: PRINT "*"
310 GOSUB 9000
330 REM -----
340 REM RUTINA PRINCIPAL
350 REM -----
360 IF XN < 0 AND XM <= 0 THEN MI
= XN
: MA = 0
370 IF XN < 0 AND XM > 0 THEN MI = XM
: MA = XM
380 IF XN >= 0 AND XM > 0 THEN MI
= 0
: MA = XM
390 IF MA > 250 THEN MA = 250
400 IF MI < - 250 THEN MI = - 250
410 K$ = "X"
: GOSUB 3000
420 GOSUB 4000
430 IF YN < 0 AND YM <= 0 THEN MI
= YN
: MA = 0
440 IF YN < 0 AND YM > 0 THEN MI = YM
: MA = YM
450 IF YN >= 0 AND YM > 0 THEN MI
= 0
: MA = YM
460 IF MA > 250 THEN MA = 250
470 IF MI < - 250 THEN MI = - 250
480 K$ = "Y"
: GOSUB 3000
490 HGR2
: HCOLOR= 3
500 GOSUB 1000
510 IF T$ = "1" THEN GOSUB 7000
: GOTO 530
520 GOSUB 5000
530 PRINT CHR$( 7)
: GET A$
540 TEXT
: HOME
: VTAB 10
: INPUT "QUIERE CAMBIAR LA FUNCION?
";A$
550 IF LEFT$( A$,1) = "N" THEN
RUN
560 PRINT
570 INVERSE
: PRINT "SUSTITUYA LA FUNCION EN
LA LINEA 100 y PULSE <RUN>."
"
: NORMAL
580 LIST 100
: END
997 REM -----
998 REM DIBUJA EJES

```

```

999 REM -----
1000 HPLOT 0,0 TO 279,0 TO 279,191
TO 0,191 TO 0,0
1010 IF D > 191 THEN YO = 96
1020 HPLOT 15,YO TO 265,YO
: HPLOT XD,185 TO XO,5
1030 N2 = 5
: FOR N1 = 2 TO 0 STEP -1
: HPLOT XO - N1,N2 TO XO + N1,N2
: N2 = N2 -1
: NEXT N1
1040 N2 = 265
: FOR N1 = 2 TO 0 STEP - 1
: HPLOT N2,YO - N1 TO N2,YO + N1
: N2 = N2 + 1
: NEXT N1
1050 HPLOT 271,YO - 2 TO 276,YO + 3
: HPLOT 276,YO - 2 TO 271,YO + 3
1060 HPLOT XO - 10,3 TO XO - 7,6
: HPLOT XO - 4,3 TO XO -10,8
1070 IF FX = 1 THEN 1140
1080 FOR G = XO TO 15 STEP - FX
1090 HPLOT G,YO + 1 TO G,YO -1
1100 NEXT G
1110 FOR G = XO TO 265 STEP FX
1120 HPLOT G,YO + 1 TO G,YO - 1
1130 NEXT G
1140 IF FY = 1 THEN 1210
1150 FOR G = YO TO 5 STEP - FY
1160 HPLOT XO + 1,G TO XO - 1,G
1170 NEXT G
1180 FOR G = YO TO 185 STEP FY
1190 HPLOT XO + 1,G TO XO - 1,G
1200 NEXT G
1210 RETURN
1997 REM -----
1998 REM PRESENTACION
1999 REM -----
2000 A$ = "*"
2010 FOR J = 1 TO 23
2020 HTAB 1
: VTAB J
: PRINT A$
2030 NEXT J
2040 FOR J = 2 TO 39
2050 HTAB J
: VTAB 23
: PRINT A$
2060 NEXT J
2070 FOR J = 23 TO 1 STEP -1
2080 HTAB 39
: VTAB J
: PRINT A$
2090 NEXT J
2100 FOR J = 39 TO 2 STEP -1
2110 HTAB J
: VTAB 1
: PRINT A$
2120 NEXT J
2130 VTAB 1
: HTAB 10
: INVERSE
: PRINT "FUNCIONES MATEMATICAS"
: NORMAL
2140 RETURN
2997 REM -----
2998 REM REDONDEADO
2999 REM -----
3000 MAX = INT (MA)
: MIX = INT (MI)

```

```

3010 IF MA > 0 AND MAX < > (MA)
      THEN MA = MAX + 1
      : GOTO 3030
3020 MA = MAX
3030 IF MI > 0 AND MIX < > MI THEN MI
      = MIX + 1
      : GOTO 3050
3040 MI = MIX
3047 REM -----
3048 REM CALCULO FACTOR DE ESCALA
3049 REM -----
3050 D = MA - MI
3060 IF K$ = "X" THEN F = LX / D
      : GOTO 3080
3070 F = LY / D
3080 GOSUB 6000
      : REM NORMALIZA EL FACTOR DE ESCA
        LA
3090 IF MI < 0 THEN V = - MI
      : GOTO 3110
3100 V = 0
3110 IF K$ = "X" THEN FX = F * 10 ^ K
      : XO = V * FX + 15
      : RETURN
3120 FY = F * 10 ^ K
      : YO = 185 - V * FY
      : RETURN
3997 REM -----
3998 REM MAXIMO Y MINIMO
3999 REM -----
4000 DX = (XM - XN) / NP
      : X = XN
4010 YN = FN Y(X)
4020 YM = FN Y(X)
4030 FOR X = XN + DX TO XM STEP DX
4040 Y = FN Y(X)
4050 IF Y > YM THEN YM = Y
4060 IF Y < YN THEN YN = Y
4070 NEXT X
4080 REM ESCRIBE MAXIMO Y MINIMO
4900 RETURN
4997 REM -----
4998 REM GRAFICO POR SEGMENTOS
4999 REM -----
5000 X = XN
      : Y = FN Y(X)
5010 GOSUB 8000
5020 X1 = XO + X * FX
5030 Y1 = YO - Y * FY
5040 FOR X = XN + DX TO XM STEP DX
5050 Y = FN Y(X)
5060 XS = XO + X * FX
5070 YS = YO - Y * FY
5080 IF T$ = "3" THEN GOSUB 5900
      : GOTO 5100
5090 HPLOT X1,Y1 TO XS,YS
5100 X1 = XS
      : Y1 = YS
5110 NEXT X
5120 RETURN
5897 REM -----
5898 REM DIBUJA SIMBOLOS
5899 REM -----
5900 XA = SX(1) * S1 + XS
      : YA = SY(1) * S2 + YS
5910 XC = XA
      : YC = YA
5920 FOR I = 2 TO 4
5930 XB = SX(I) * S1 + XS
      : YB = SY(I) * S2 + YS

```

```

5940 HPLOT XA,YA TO XB,YB
5950 XA = XB
      : YA = YB
5960 NEXT I
5980 HPLOT XB,YB TO XC,YC
5990 RETURN
5997 REM -----
5998 REM NORMALIZACION
5999 REM -----
6000 K = 0
      : F = INT (F)
6010 IF F > V1 THEN F = F / 10
      : K = K + 1
      : GOTO 6010
6020 L = 0
6030 FOR I = 1 TO 10
6040 IF L < > 0 OR VN(I) = 0 THEN 6070

6050 IF F < VN(I) THEN L = I - 1
6060 IF F = VN(I) THEN L = I
6070 NEXT I
6080 F = VN(L)
6090 RETURN
6997 REM -----
6998 REM GRAFICO POR PUNTOS
6999 REM -----
7000 GOSUB 8000
7010 FOR X = XN TO XM STEP DX
7020 Y = FN Y(X)
7030 XS = XO + X * FX
7040 YS = YO - Y * FY
7050 HPLOT XS,YS
7060 NEXT X
7070 RETURN
7997 REM -----
7998 REM ESCALA PARA SIMBOLOS
7999 REM -----
8000 S1 = INT (FX /.4)
      : IF S1 < 1 THEN S1 = 1
8010 S2 = INT (FY / 4)
      : IF S2 < 1 THEN S2 = 1
8020 IF S1 < S2 THEN S2 = S1
      : RETURN
8030 S1 = S2
8040 RETURN
8997 REM -----
8998 REM PAUSA
8999 REM -----
9000 HOME
      : HTAB 12
      : VTAB 12
      : FLASH
      : PRINT "UN MOMENTO, POR FAVOR"
      : NORMAL
      : RETURN
9997: REM -----
9998 REM GESTION DE ERRORES
9999 REM -----
10000 ER = PEEK (222)
      LOC = PEEK (218) + PEEK (219)
      * 256
10010 IF ER < > 53 AND ER < > 133
      AND ER < > 254 AND ER < > 69
      AND ER < > 191 THEN HOME
      : PRINT "ERROR N.",ER
      : END
10020 IF ER = 191 THEN HOME
      : HTAB 12
      : VTAB 12
      : INVERSE

```

```

: PRINT "FORMULA DEMASIADO COMPLEJA"
: NORMAL
: STOP
10030 IF ER = 254 THEN RUN
10040 REM CANTIDAD ILEGAL
10050 IF XS > 279 THEN XS = 279
: RESUME
10060 IF XS < 0 THEN XS = 0
: RESUME
10070 IF YS > 191 THEN YS = 191
: RESUME
10080 IF YS < 0 THEN YS = 0
: RESUME
10090 IF XA > 279 THEN XA = 279
: RESUME
10100 IF XA < 0 THEN XA = 0
: RESUME
10110 IF YA > 191 THEN YA = 191
: RESUME
10120 IF YA < 0 THEN YA = 0
: RESUME
10130 IF XB > 279 THEN XB = 279
: RESUME
10140 IF XB < 0 THEN XB = 0
: RESUME
10150 IF YR > 191 THEN YR = 191
: RESUME
10160 IF YR < 0 THEN YR = 0
: RESUME
10170 IF X1 > 279 THEN X1 = 279
: RESUME
10180 IF X1 < 0 THEN X1 = 0
: RESUME
10190 IF Y1 < 191 THEN Y1 = 191
: RESUME
10200 IF Y1 < 0 THEN Y1 = 0
: RESUME
10210 IF XC > 279 THEN XC = 279
: RESUME
10220 IF XC < 0 THEN XC = 0
: RESUME
10230 IF YC > 191 THEN YC = 191
: RESUME
10240 IF YC < 0 THEN YC = 0
: RESUME
10250 IF SW = 1 THEN 10300
10260 TEXT
: HOME
: VTAB 12
: INVERSE
: PRINT "LA FUNCION NO ESTA DEFINIDA
EN ALGUNOS PUNTOS DEL INTERVA
LO."
: NORMAL
10270 VTAB 23
: PRINT " PULSE UNA TECLA PARA CONTI
NUAR ";
: GET A$
10280 GOSUB 9000
10290 SW = 1
10300 X = X + DX
: IF X > XM THEN 10320
10310 RESUME
10320 IF LOC = 4010 THEN 4020
10330 IF LOC = 4020 THEN 4030
10340 IF LOC = 4040 THEN 430
10350 IF LOC = 3000 THEN 5010
10360 IF LOC = 5050 THEN 530
10370 IF LOC = 7020 THEN 530

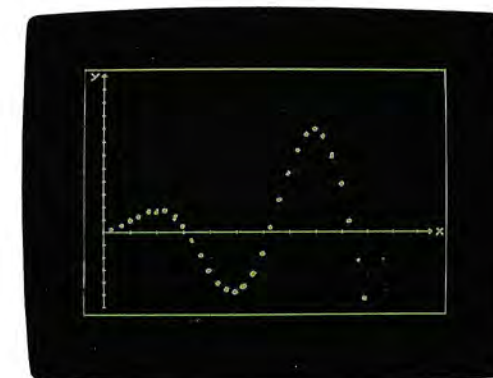
```

PRESENTACION DEL GRAFICO DE UNA FUNCION

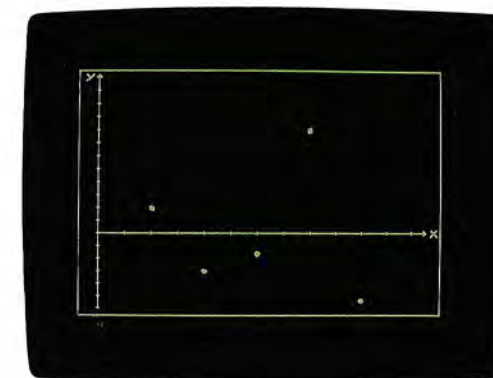
Al ejecutar el programa, después de la parte de inicialización (líneas 100 a 180), se llama la subrutina 2000 (línea 220), que presenta el recuadro de asteriscos y la escritura invertida. Al final de la 2000, el programa vuelve al main y presenta, en orden, las diversas posibilidades. La foto muestra la situación final después de haber contestado todas las preguntas y antes de introducir la selección del tipo de gráfico. Los datos introducidos se refieren a la función $X * \text{SEN}(X)$ (línea 100). Para la misma función, las fotos que siguen muestran los diversos aspectos del gráfico en función de los parámetros NUMERO DE PUNTOS y TIPO DE GRAFICO.



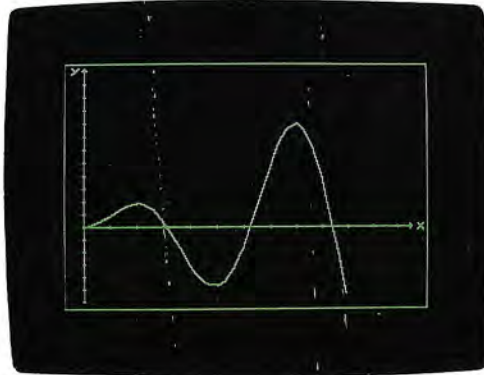
Se ha elegido la presentación del gráfico por puntos (tipo de gráfico 1). El programa presenta los ejes cartesianos y los puntos representativos de los treinta pares de coordenadas. El valor de X es introducido por el usuario, mientras que el valor de Y lo calcula el programa.



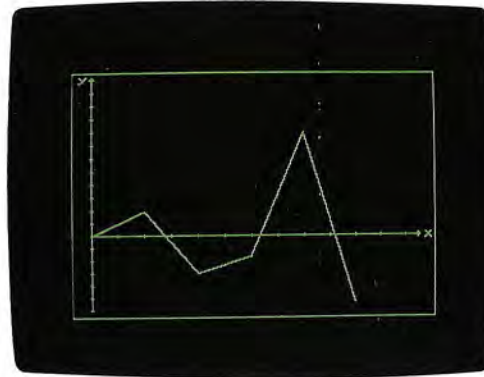
En este caso, el usuario ha elegido la presentación del gráfico por puntos sobre la base de un número reducido de valores de la variable X (5). El proceso de la función siempre es el mismo, pero resulta mucho menos reconocible.



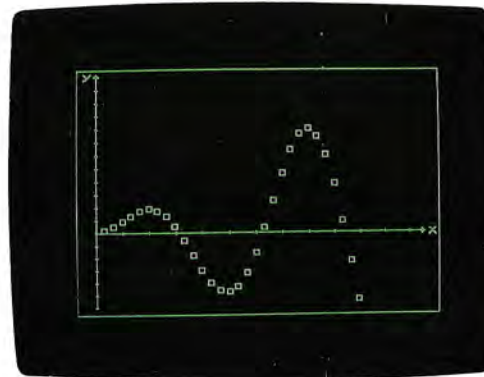
Presentación por segmentos sobre la base de 30 valores de la variable X. Dado el elevado número de datos, la definición del proceso es excelente. El programa ha trazado una serie de segmentos que unen de dos en dos los puntos de coordenadas conocidas.



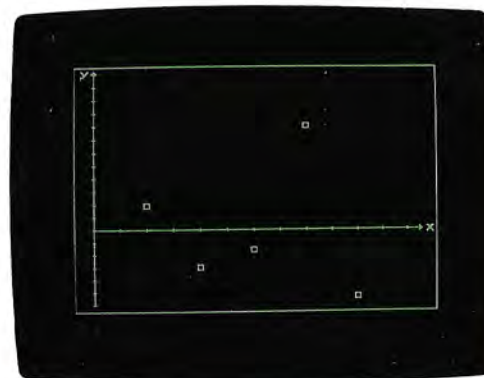
La presentación es aún por segmentos, pero sobre la base de 5 valores de X en la entrada. La definición del gráfico es muy aproximada, y se distinguen claramente los segmentos.



El usuario ha elegido la presentación por símbolos sobre la base de 30 valores de la variable X. Esta representación es útil cuando debe presentarse el proceso de una magnitud sujeta a errores de aproximación. La amplitud del símbolo empleado puede representar en este caso el error.



Presentación por símbolos sobre la base de 5 valores de la variable independiente. El proceso de la función es escasamente reconocible.



Al final de la presentación del gráfico, pulsando una tecla cualquiera (línea 540), el programa presenta la pregunta de variación de función. Obsérvese la línea 580, que permite la presentación del contenido de la línea 100 como recordatorio para evidenciar la función existente. A la 580 le sigue la instrucción END que termina el programa. De esta manera, el programa vuelve al estado de comandos (la nueva condición es evidenciada por la aparición del prompt) y el usuario puede sustituir la función reescribiendo la línea 100. Al final, el comando RUN activa la ejecución del gráfico con la nueva función $X/(X-2)$.

El programa presenta aún al usuario las peticiones de los parámetros necesarios para la ejecución.

A continuación de la elección del tipo de gráfico, el programa pasa a la fase de cálculo, durante la cual descubre que la función tiene un punto de discontinuidad para $X = 2$. En efecto, para este valor se tiene $Y = 2/(2-2) = 2/0$. El cociente $2/0$ proporciona como resultado un número infinito, que no puede ser representado en el gráfico. El programa detecta esta anomalía (no se trata de un error, sino de una característica de la función) y la indica. El diagnóstico que aparece es muy general y a veces puede resultar inexacto desde el punto de vista matemático. Esto se debe a la estructura de la subrutina de error, que acumula varias causas en el mismo mensaje.

La función se presenta después de que el usuario ha dado el consentimiento pulsando una tecla cualquiera (línea 10250 y siguientes). Es evidente la presencia de una asíntota para $X = 2$ (punto de discontinuidad de la función). El proceso puede detallarse con mayor definición pidiendo la presentación por segmentos.

```

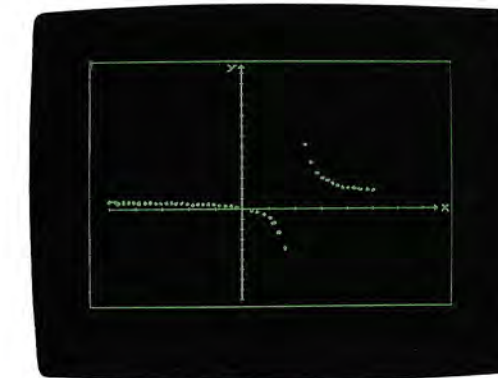
QUIERE CAMBIAR LA FUNCION? SI
SUSTITUYA LA FUNCION EN LA LINEA 100
Y PULSE <RUN>
100 DEF FN Y(X) = X * SEN (X)
1100 DEF FN Y(X) = X / (X-2)
IRUN ■
    
```

```

*****FUNCIONES MATEMATICAS*****
INTERVALO EJE X? -5,5
NUMERO DE PUNTOS? 50
ELIJA EL TIPO DE GRAFICO: ■
1) POR PUNTOS
2) POR SEGMENTOS
3) POR SIMBOLOS
    
```

```

LA FUNCION NO ESTA DEFINIDA EN ALGUNOS PUNTOS
DEL INTERVALO.
PULSE UNA TECLA PARA CONTINUAR ■
    
```



Presentación de caracteres en el modo gráfico

También las aplicaciones gráficas requieren a menudo la introducción de caracteres numéricos o alfabéticos en la imagen vídeo. Por ejemplo, cuando en un gráfico quieren insertarse textos explicativos. Como en el empleo del monitor de alta resolución cada punto de la pantalla se gestiona por separado, ya no es posible presentar los caracteres ASCII estándar que se emplean en los textos. Sin embargo, en esta última modalidad de funcionamiento, el vídeo está gestionado por matrices de carácter, en el sentido de que en ellas es posible localizar un carácter en una de las «casillas» que se encuentran en el cruce de cada línea con cada columna. Cada casilla, a su vez, está constituida por una matriz de puntos de pantalla, que el sistema activa automáticamente según el carácter que debe visualizarse.

En la modalidad gráfica, la correspondencia carácter/configuración ya no existe, precisamente para permitir la gestión independiente de todos los puntos de la pantalla.

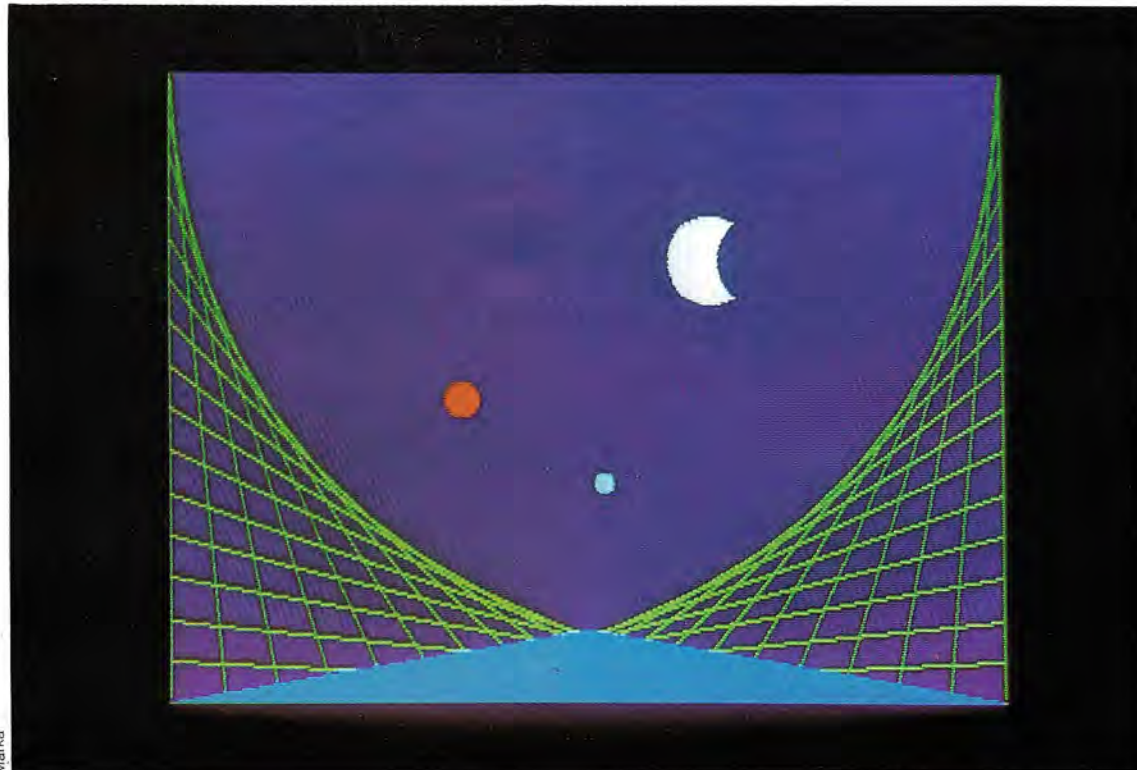
Por tanto, el vídeo puede emplearse para pre-

sentar textos (modo texto) o gráficos (modo gráfico), y las dos modalidades de funcionamiento no pueden coexistir. En algunas máquinas se establece un compromiso reservando algunas líneas (generalmente las 2 o 3 líneas de más abajo) para texto cuando la parte restante de la pantalla está gestionada en el modo gráfico y, por tanto, no puede presentar caracteres.

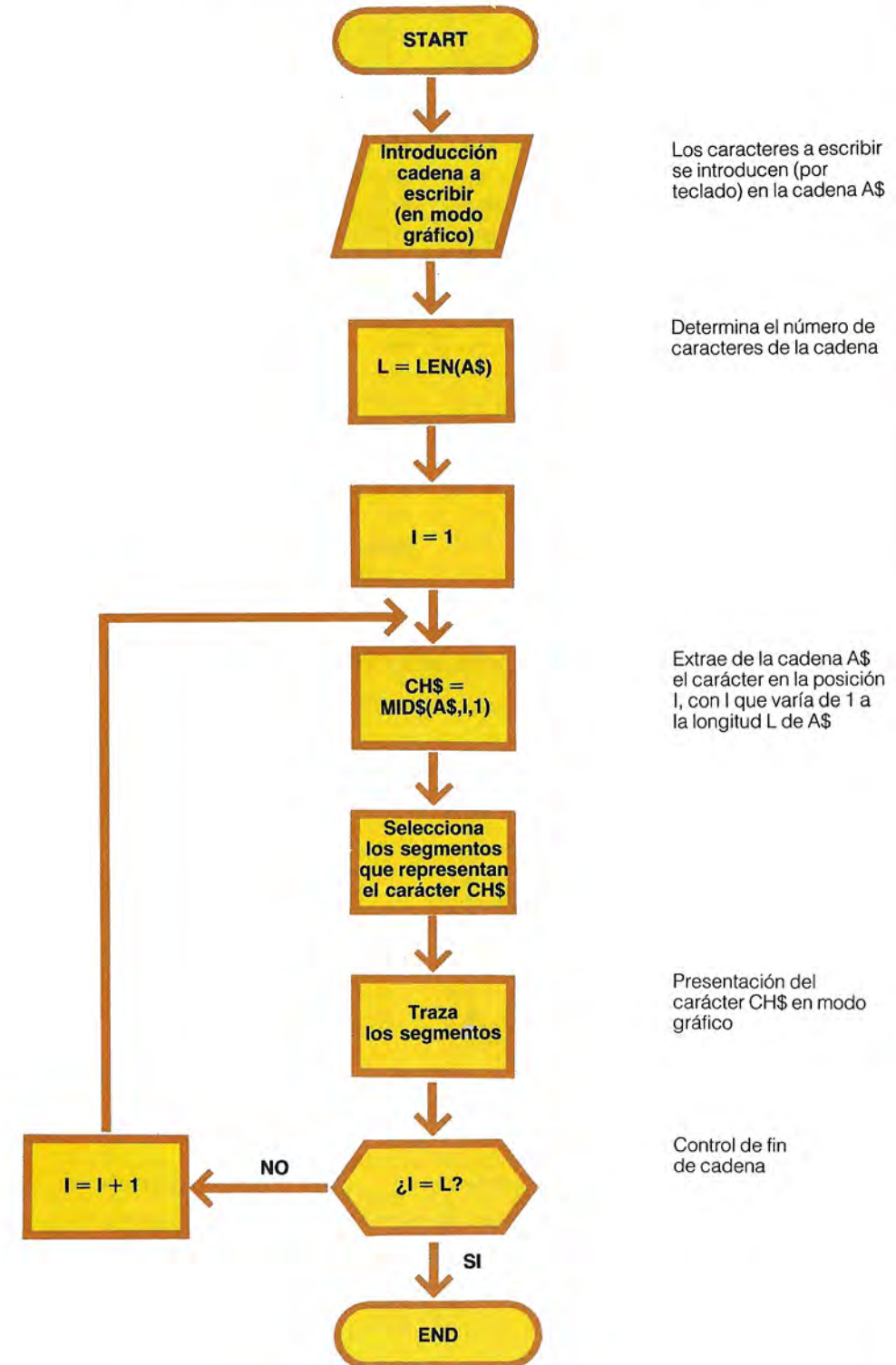
Si se desea presentar un carácter alfabético o numérico en una página gráfica es necesario construir punto por punto el carácter por software. En la mayoría de ordenadores personales y microordenadores no hay previstas subrutinas de sistema que puedan realizar esta tarea y, por tanto, el usuario debe escribirlas por sí mismo. La técnica consiste en representar las letras y los números de manera esquematizada mediante desplazamientos elementales del punto de escritura. Para obtener una leyenda cualquiera en modo gráfico es necesario activar, letra por letra o número por número, la correspondiente rutina. La lógica empleada se ha representado en la figura de la página siguiente.

El programa no presenta ninguna dificultad, excepto en la parte de selección de los segmentos que componen cada carácter a representar.

Imagen gráfica obtenida utilizando funciones matemáticas.



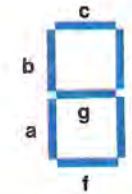
PRESENTACION DE CARACTERES EN MODO GRAFICO



El modo más sencillo de proceder sería escribir una rutina para cada carácter: para escribir la letra «a» debería llamarse la primera rutina, para la «b» la segunda, y así sucesivamente. Sin embargo, este método tiene el defecto de no ser optimizado. De hecho, muchos caracteres gráficos tienen una forma similar. Por ejemplo, la letra F y la letra E pueden obtenerse una de otra añadiendo o retirando un trazo horizontal. Para otros caracteres, esta característica es menos evidente, pero siempre es posible esquematizar el conjunto de caracteres para obtener un equivalente bastante válido de cada uno de ellos compuesto sólo de segmentos. Como se sabe, este método es muy utilizado para la representación de números en los presentadores numéricos. En la figura de abajo se ha representado el conjunto de cifras de 0 a 9 que puede obtenerse utilizando siete segmentos, cada uno de los cuales está indicado con una letra del alfabeto (a,b,c,d,e,f,g); activando selectivamente los segmentos se tiene la presentación de cada una de las diez cifras. Por ejemplo, si todos los segmentos son visibles

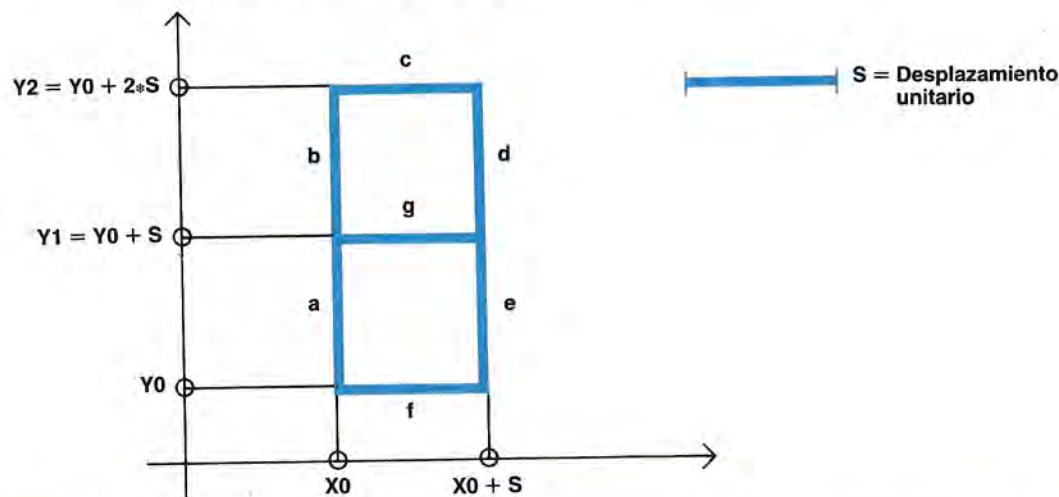
(como en la figura), se tiene la presentación de la cifra 8; apagando el segmento g se obtiene el 0; apagando los segmentos f, a, b y c se obtiene el número 1, y así sucesivamente. En la primera columna de la tabla de la página siguiente se han representado las cifras de 1 a 0, en la segunda los segmentos que deben activarse y en la tercera el símbolo que aparece. En la última columna se han representado los segmentos a activar utilizando la simbología 0/1 en lugar de la notación SI/NO, que permite ocupar un menor espacio de memoria y realizar más rápidamente las selecciones. Por ejemplo, para activar la escritura del número 6 son necesarios los segmentos g, f, e, b y a. En el programa, cada segmento activado está representado con un 1 (SI), mientras que los desactivados se indican con un 0; por tanto, para el número 6, la simbología será 1 1 1 0 0 1 1. Para presentar un determinado número debe controlarse el contenido de la matriz posición por posición, y cuando se encuentra el valor 1 debe llamarse la rutina que activa el segmento correspondiente.

REPRESENTACION DE LAS CIFRAS CON SIETE SEGMENTOS



Número a representar	Segmento activado							Símbolo representado	Contenido de la matriz
	7 g	6 f	5 e	4 d	3 c	2 b	1 a		
1	—	—	SI	SI	—	—	—	1	0 0 1 1 0 0 0
2	SI	SI	—	SI	SI	—	SI	2	1 1 0 1 1 0 1
3	SI	SI	SI	SI	SI	—	—	3	1 1 1 1 1 0 0
4	SI	—	SI	—	—	SI	—	4	1 0 1 0 0 1 0
5	SI	SI	SI	—	SI	SI	—	5	1 1 1 0 1 1 0
6	SI	SI	SI	—	—	SI	SI	6	1 1 1 0 0 1 1
7	—	—	SI	SI	SI	—	—	7	0 0 1 1 1 0 0
8	SI	SI	SI	SI	SI	SI	SI	8	1 1 1 1 1 1 1
9	SI	—	SI	SI	SI	SI	—	9	1 0 1 1 1 1 0
0	—	SI	SI	SI	SI	SI	SI	0	0 1 1 1 1 1 1

REPRESENTACION DE LAS CIFRAS DE SIETE SEGMENTOS



Segmento	Coordenadas		Incrementos	
	de	a	de	a
a	X0, Y0	X0, Y0 + S	0,0	0,1
b	X0, Y0 + S	X0, Y0 + 2 * S	0,1	0,2
c	X0, Y0 + 2 * S	X0 + S, Y0 + 2 * S	0,2	1,2
d	X0 + S, Y0 + 2 * S	X0 + S, Y0 + S	1,2	1,1
e	X0 + S, Y0 + S	X0 + S, Y0	1,1	1,0
f	X0 + S, Y0	X0, Y0	1,0	0,0
g	X0, Y0 + S	X0 + S, Y0 + S	0,1	1,1

Por tanto, para cada uno de los 7 segmentos es necesario preparar la rutina adecuada. En realidad no son necesarias siete rutinas diferentes: basta con prever una sola, parametrizada. En la segunda columna de la tabla de la página anterior se han representado los desplazamientos, referidos al punto de origen de coordenadas X0, Y0, necesarios para obtener cada segmento. Por ejemplo, el segmento a puede obtenerse con un solo desplazamiento de X0, Y0 a X0, Y0 + S, donde S indica la longitud de cada segmento. Adoptando esta disposición (desplazamiento unitario de S) se obtienen dos importantes parametrizaciones:

desplazándose desde el punto X0, Y0 al punto X0, Y0 + S, la primera posición (X0, Y0) no tiene variaciones (incrementos) respecto al origen; es decir, su incremento es 0 para el eje X y 0 para el eje Y. Y viceversa, las coordenadas finales del segmento (X0 + S, Y0 + S) tienen 0 como incremento en el eje X y S como incremento en el eje Y. Utilizando esta parametrización puede escribirse una rutina generalizada que, empleando los desplazamientos indicados en la figura de enfrente, puede trazar todos los segmentos. Indicando con X1, Y1 el punto de inicio de un segmento general y con X2, Y2 su punto final, se tiene:

- Todos los desplazamientos, y por tanto todos los segmentos, no se indican con coordenadas absolutas, sino relativas y un punto inicial; para desplazar la posición de la presentación del carácter basta con variar las coordenadas del punto de origen.
- Las dimensiones de los caracteres pueden variarse sólo con variar el parámetro S.

$$X1 = X0 + N_x S \text{ (incremento eje X)}$$

$$Y1 = Y0 \pm N_y S \text{ (incremento eje Y)}$$

y los análogos para X2, Y2 (recuérdese que X0, Y0 son las coordenadas de referencia). En el caso particular, para los primeros incrementos se tiene N = 0 (punto de inicio), y por tanto:

$$X1 = X0$$

$$Y1 = Y0$$

Por ejemplo, para el segmento a, que se obtiene

Para el segundo punto, en cambio, se tiene $N_x = 0$ y $N_y = 1$, por lo que

$$X_2 = X_0$$

$$Y_2 = Y_0 \pm S$$

X_1, Y_1 y X_2, Y_2 son las coordenadas de los extremos. El signo a adoptar en la segunda expresión (+ o -) depende de la orientación del eje Y. En las máquinas en que el origen de la pantalla de vídeo está abajo, el signo es + (las Y crecen hacia arriba), mientras que en las que tienen el origen arriba el signo es -.

Para obtener una rutina generalizada basta con memorizar los incrementos (ver la figura de la pág. 1484) de cada segmento y los segmentos (figura de la pág. anterior) que componen cada número.

En la figura de enfrente se ha representado el diagrama de flujo de un programa de demostración que presenta un número de las dimensiones deseadas (mediante el parámetro S) y en cualquier posición de la pantalla (variando el origen X_0, Y_0). Los incrementos necesarios para obtener cada segmento se memorizan en cuatro variables dimensionadas:

$XD(l), YD(l)$ = coordenadas del punto inicial de un segmento
 $XA(l), YA(l)$ = coordenadas del punto final de un segmento

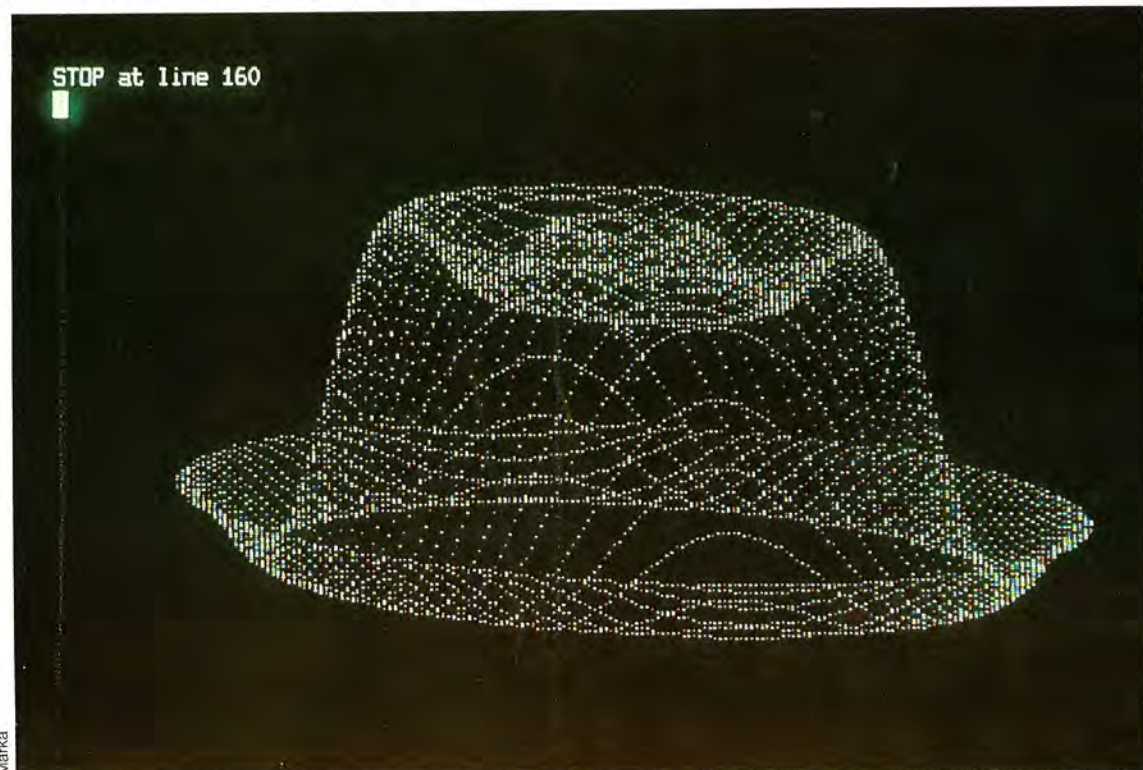
Los valores se asignan a la línea 60 con los DATA de las líneas 910 a 916 (compárense estos DATA con la tabla de la pág. 1484).

En cambio, los segmentos a visualizar para obtener cada una de las diez cifras se memorizan en la variable de dos dimensiones $C\%(10,7)$ y se asignan con la READ de la línea 76 y los correspondientes DATA de las líneas 918 a 927 (véase la tabla de la pág. 1485). La primera dimensión (10) indica una de las diez posibles cifras a representar (0,1,2...); la otra dimensión (7) memoriza los segmentos a activar.

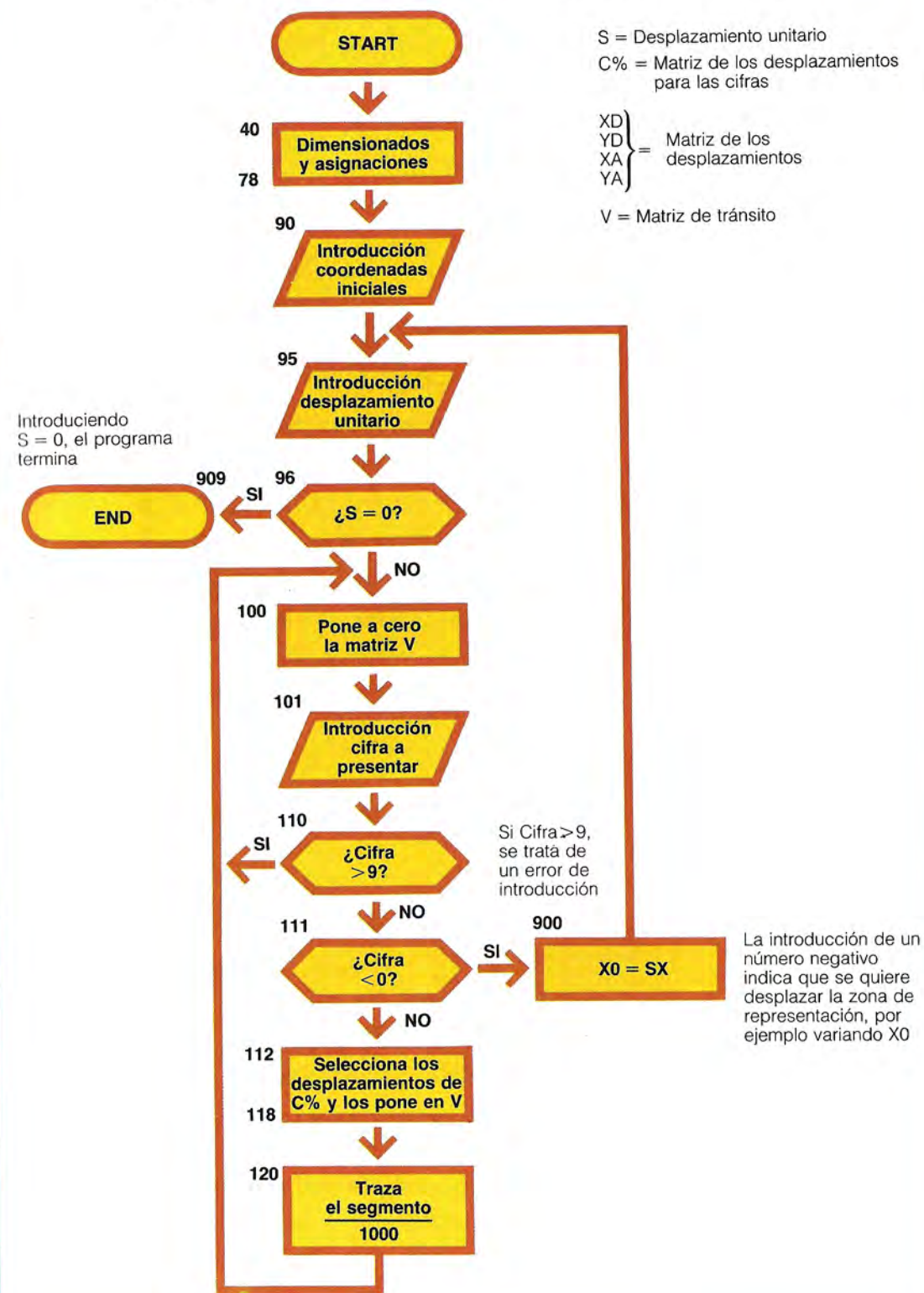
En el programa se utiliza una matriz de apoyo (V, con dimensión 7) a la que se transfieren los elementos de $C\%$ correspondientes a los segmentos a representar (línea 116). Por tanto, el control se realiza en esta matriz.

En la subrutina 1000 se llama la 2000 (figura de abajo de la pág. 1488) que calcula las posiciones de principio (X_1, Y_1) y de final (X_2, Y_2) del segmento y lo traza.

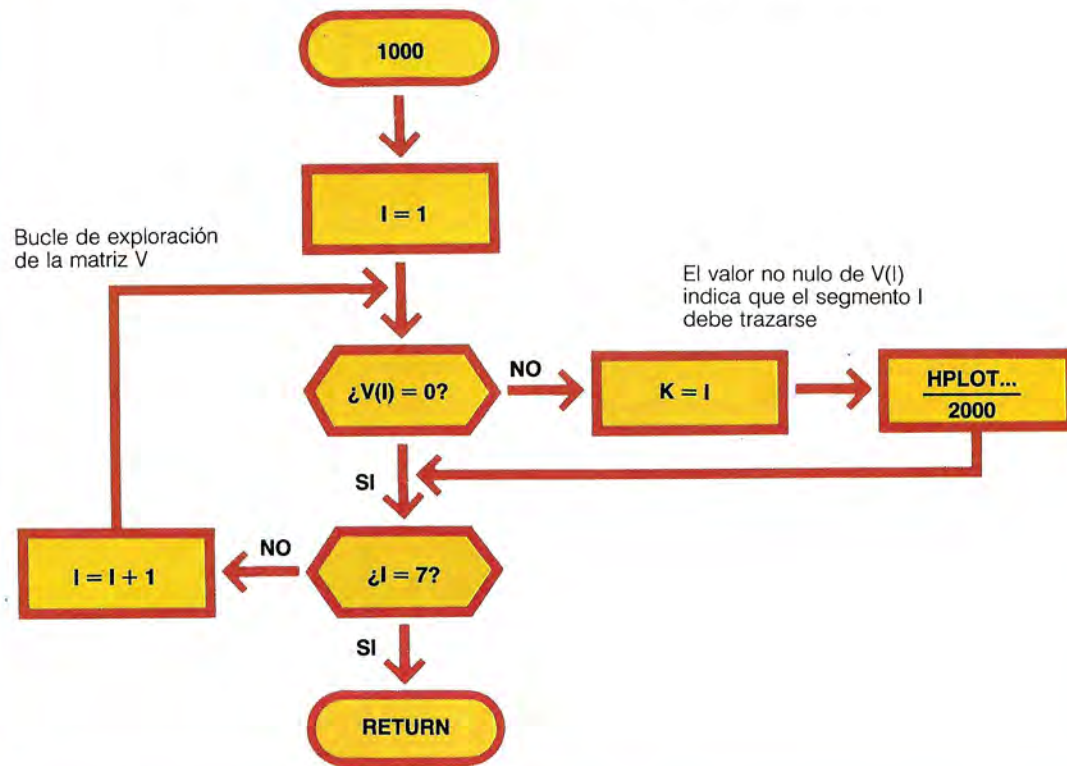
Gráfico tridimensional para curvas de nivel.



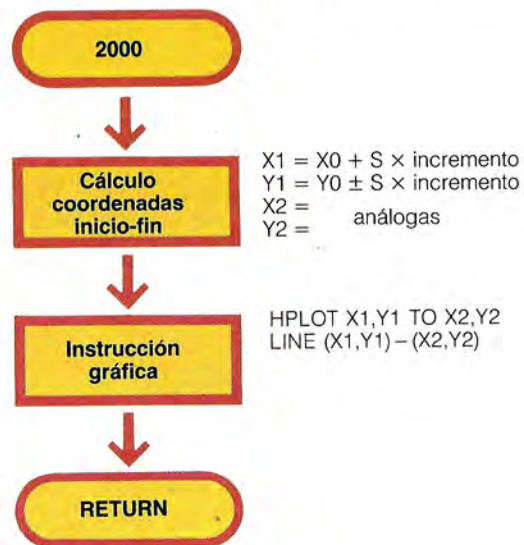
PRESENTACION DE UN NUMERO EN MODO GRAFICO



SUBROUTINA DE PRESENTACION DE UN SEGMENTO



SUBROUTINA DE TRAZADO



Esta subrutina depende del tipo de máquina que se utilice

Si se utilizan otras máquinas (el programa del listado de esta página y de la siguiente, corre en sistemas Apple y Siprel 2010), basta con modificar únicamente la subrutina 3000. Así por ejemplo, en la versión para Olivetti M20, las líneas 3010, 3030 y 3040 se convierten en:

```

3010 Y1 = Y0 + YD (K) * S
3030 Y2 = Y0 + YA (K) * S
3040 LINE (X1,Y1) - (X2,Y2)
  
```

Todo el resto puede utilizarse íntegramente. También el problema de la presentación de los caracteres alfabéticos puede resolverse ágilmente escribiendo una rutina que presente cada letra como si se tratase de un dibujo, o sea

esquematisando los caracteres con una serie de segmentos de manera análoga a la adoptada para los números. Para los números es posible definir 7 segmentos de base que, apagados o encendidos en la sucesión adecuada, generan todos los números. Para las letras, una esquematización de este tipo conduciría a definir demasiados segmentos de base. En lugar de ello conviene utilizar otro método, muy similar, que permite generar una figura cualquiera. Cada letra se esquematiza con una serie de segmentos (para un máximo de 11 en esta aplicación) y su presentación se obtiene posicionando el haz electrónico en correspondencia con el punto elegido como inicio y trazando a continuación los segmentos necesarios.

PRESENTACION DE CIFRAS EN MODO GRAFICO

```

10 REM
20 REM : NUMEROS
30 REM ..
40 DIM XD(7),YD(7),XA(7),YA(7)
45 DIM V(7),C%(10,7)
46 HGR
50 FOR I = 1 TO 7
60 READ XD(I),YD(I),XA(I),YA(I)
70 NEXT I
72 FOR I = 1 TO 10
74 FOR J = 7 TO STEP - 1
76 READ C%(I,J)
: NEXT J
78 NEXT I
90 PRINT "Coordenadas iniciales"
91 INPUT "X0 = ?";X0
92 INPUT "Y0 = ?";Y0
93 SX = X0
95 INPUT "Desplazamiento unitario ? ";S

96 IF S = 0 GOTO 909
97 Y0 = Y0 + 3 * 5
100 FOR I = 1 TO 7
: V(I) = 0
: NEXT I
101 INPUT "NUMERO ?";N
110 IF N > 9 GOTO 100
111 IF N < 0 GOTO 900
112 LX = N
IF LX = 0 THEN LX = 10
114 FOR I = 1 TO 7
116 V(I) = C%(LX,I)
118 NEXT I
120 GOSUB 1000
160 X0 = X0 + 2 * 5
170 GOTO 100
900 X0 = SX
902 GOTO 95
909 END
  
```

```

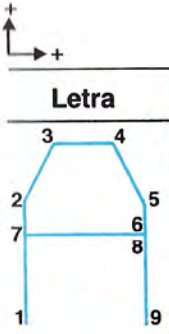
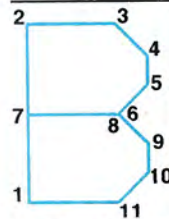
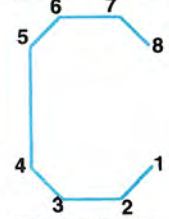
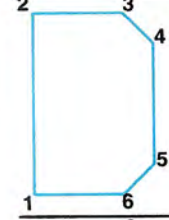
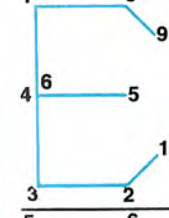
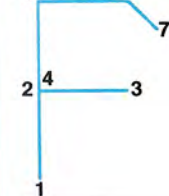
910 DATA 0,0,0,1
: REM =a
911 DATA 0,1,0,2
: REM =b
912 DATA 0,2,1,2
: REM =c
913 DATA 1,2,1,1
: REM =d
914 DATA 1,1,1,0
: REM =e
915 DATA 1,0,0,0
: REM =f
916 DATA 0,1,1,1
: REM =g
918 DATA 0,0,1,1,0,0,0
: REM =1
919 DATA 1,1,0,1,1,0,1
: REM =2
920 DATA 1,1,1,1,1,0,0
: REM =3
921 DATA 1,0,1,0,0,1,0
: REM =4
922 DATA 1,1,1,0,1,1,0
: REM =5
923 DATA 1,1,1,0,0,1,1
: REM =6
924 DATA 0,0,1,1,1,0,0
: REM =7
925 DATA 1,1,1,1,1,1,1
: REM =8
926 DATA 1,0,1,1,1,1,0
: REM =9
927 DATA 0,1,1,1,1,1,1
: REM =0
1000 FOR I = 1 TO 7
1020 IF V(I) = 0 GOTO 1040
1025 K = I
1030 GOSUB 3000
1040 NEXT I
1050 RETURN
3000 X1 = X0 + XI(K) * 5
3010 Y1 = Y0 + YI(K) * 5
3020 X2 = X0 + XI(K) * 5
3030 Y2 = Y0 + YI(K) * 5
3040 HPL01 X1,Y1 TO X2,Y2
3050 RETURN

```

En la tabla de enfrente se han representado los desplazamientos que permiten presentar los caracteres alfabéticos. Cada letra está contenida en un rectángulo de base 4 y altura 6: su posición está referida al vértice inferior izquierdo, y el sentido de los desplazamientos es positivo hacia la derecha para el eje X y positivo hacia arriba para el eje Y. La primera columna de la tabla (posicionado) indica el desplazamiento a efectuar para posicionarse sobre el primer punto, donde empieza después la serie de desplazamientos que permiten presentar el carácter.

Este desplazamiento es necesario porque la posición del carácter debe definirla el usuario. Por ejemplo, para la letra A, el trazado empieza en el mismo punto de referencia (punto 1) y, por tanto, el desplazamiento inicial es nulo; y viceversa, la C se traza empezando desde otro punto, y necesita un desplazamiento inicial (4,1). Al examinar las tablas debe recordarse que los números (1, 2..., 11) que identifican los desplazamientos no son iguales a los indicados para cada letra, los cuales sólo sirven para evidenciar el recorrido efectuado.

PRESENTACION DE CARACTERES ALFABETICOS EN MODO GRAFICO

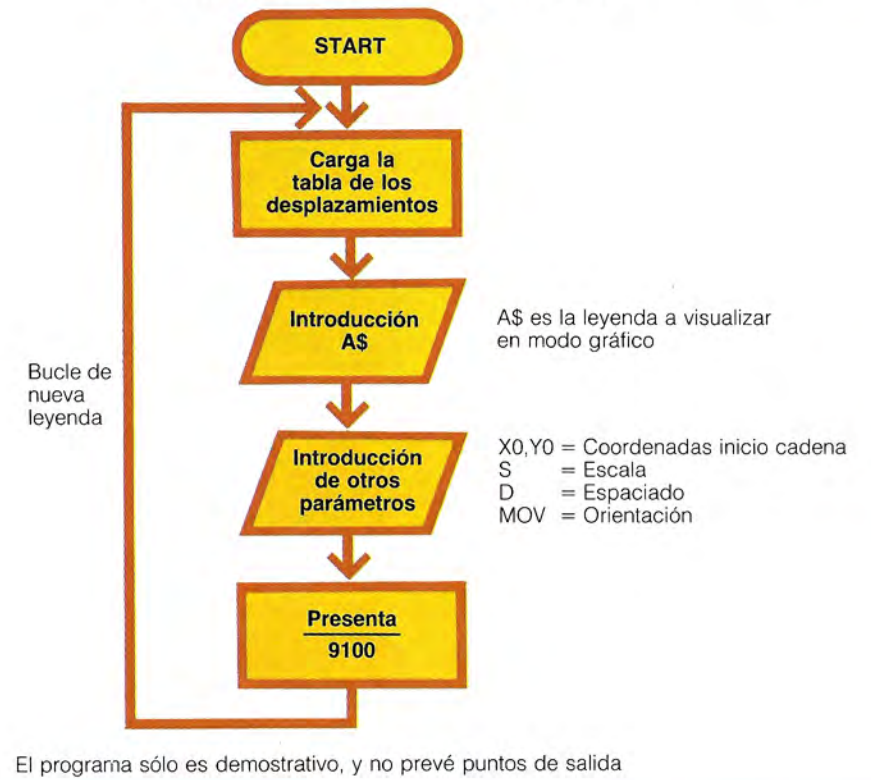
		Desplazamientos										
Letra	Posición	1	2	3	4	5	6	7	8	9	10	11
	0,0	0,4	1,2	2,0	1,-2	0,-1	-4,0	4,0	0,-3	-	-	-
	0,0	0,6	3,0	1,-1	0,-1	-1,-1	-3,0	3,0	1,-1	0,-1	-1,-1	-3,0
	4,1	-1,-1	-2,0	-1,1	0,4	1,1	2,0	1,-1	-	-	-	-
	0,0	0,6	3,0	1,-1	0,-4	-1,-1	-3,0	-	-	-	-	-
	4,1	-1,-1	-3,0	0,+3	3,0	-3,0	0,3	3,0	1,-1	-	-	-
	0,0	0,3	3,0	-3,0	0,3	3,0	1,-1	-	-	-	-	-

		1	2	3	4	5	6	7	8	9	10	11
	2,3	2,0	0,-2	-1,-1	-2,0	-1,1	0,4	1,1	2,0	1,-1	-	-
	0,0	0,6	0,-3	4,0	0,3	0,-6	-	-	-	-	-	-
	2,0	0,6	-	-	-	-	-	-	-	-	-	-
	0,2	0,-1	1,-1	1,0	1,1	0,5	1,0	-3,0	-	-	-	-
	0,0	0,6	0,-3	2,0	2,3	-2,-3	2,-3	-	-	-	-	-
	0,6	0,-6	3,0	1,1	-	-	-	-	-	-	-	-
	0,0	0,6	2,-3	2,3	0,-6	-	-	-	-	-	-	-

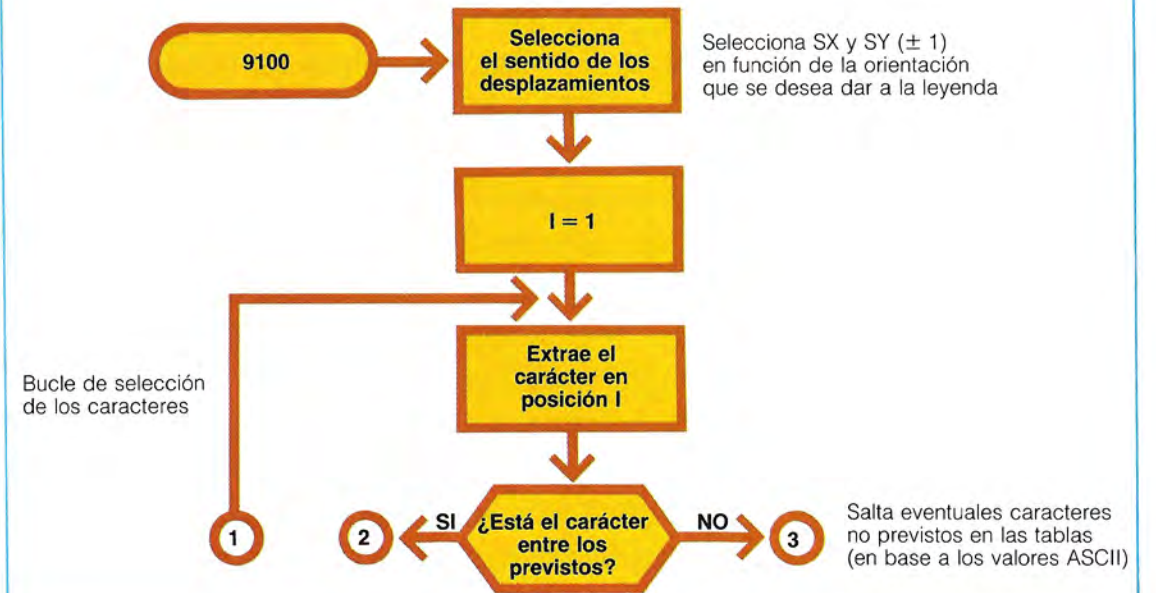
		1	2	3	4	5	6	7	8	9	10	11
	0,0	0,6	4,-6	0,6	-	-	-	-	-	-	-	-
	0,1	0,4	1,1	2,0	1,-1	0,-4	-1,-1	-2,0	-1,1	-	-	-
	0,0	0,6	3,0	1,-1	0,-1	-1,-1	-3,0	-	-	-	-	-
	2,2	1,-2	-2,0	-1,1	0,4	1,1	2,0	1,-1	0,-4	-1,-1	-	-
	0,0	0,6	3,0	1,-1	0,-1	-1,-1	-3,0	2,0	2,-3	-	-	-
	0,1	1,-1	2,0	1,1	0,1	-1,1	-2,0	-1,1	0,1	1,1	2,0	1,-1
	2,0	0,6	-2,0	4,0	-	-	-	-	-	-	-	-

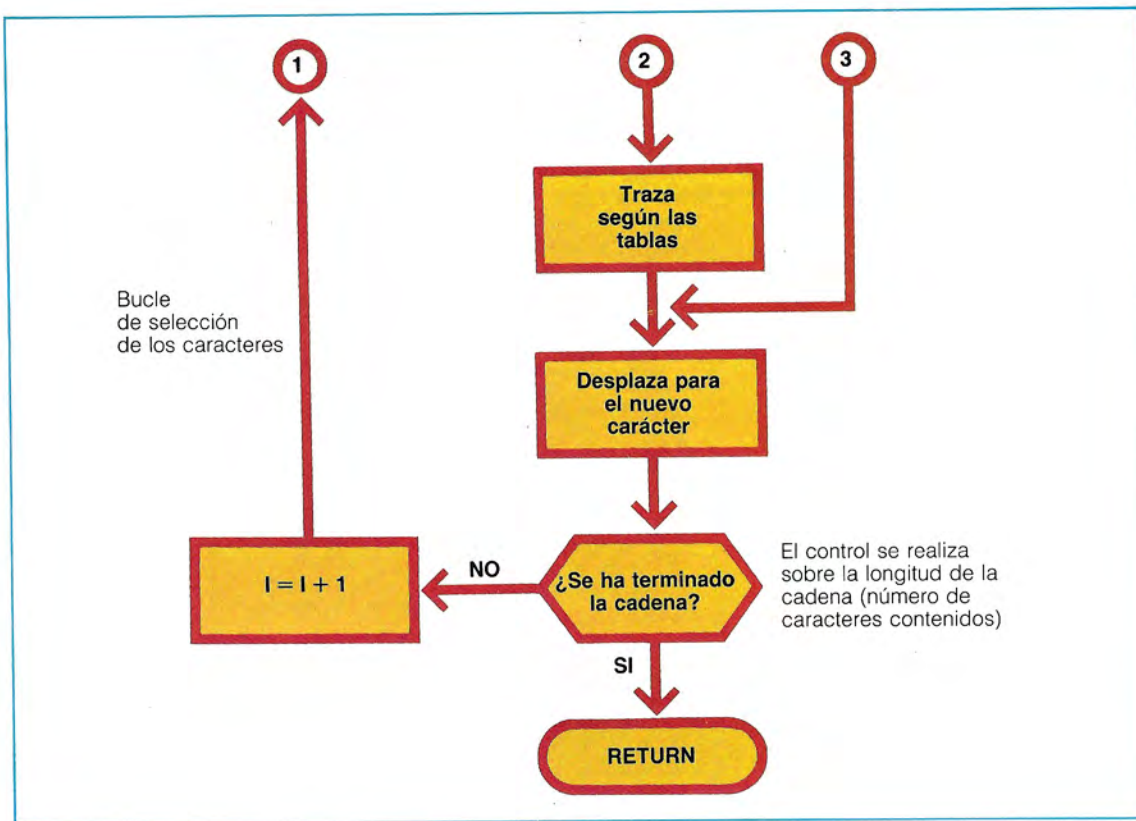
		1	2	3	4	5	6	7	8	9	10	11
	0,6	0,-4	1,-2	2,0	1,2	0,4	-	-	-	-	-	-
	0,6	0,-3	2,-3	2,3	0,3	-	-	-	-	-	-	-
	0,0	0,1	4,4	0,1	0,-1	-2,-2	-2,2	0,1	0,-1	4,-4	0,-1	-
	2,0	0,3	-2,3	2,-3	2,3	-	-	-	-	-	-	-
	0,6	0,-6	2,3	2,-3	0,6	-	-	-	-	-	-	-
	4,1	-1,-1	-3,0	4,6	-3,0	-1,-1	-	-	-	-	-	-

DIAGRAMA DE FLUJO DEL PROGRAMA PARA LA PRESENTACION DE LEYENDAS EN MODO GRAFICO



SUBROUTINA DE PRESENTACION DE LOS CARACTERES





PRESENTACION DE CARACTERES EN MODO GRAFICO

```

10 HGR
: HCOLOR= 3
20 DIM POX(26,2),CARX(26,11,2)
30 GOSUB 35Q
40 HOME
50 VTAB 22
60 INPUT A$
70 INPUT X0,Y0,S,D,MOV
80 GOSUB 9100
90 GOTO 40
320 REM -----
330 REM INICIALIZACION
340 REM -----
350 FOR I = 1 TO 26
360 READ POX(I,1),POX(I,2)
370 FOR J = 1 TO 11
380 READ CARX(I,J,1),CARX(I,J,2)
390 NEXT
400 NEXT
410 RETURN
420 REM -----
430 REM DATA
440 REM -----
450 REM A
460 DATA 0,0
470 DATA 0,4,1,2,2,0,1,-2,0,-1,-4,0,
480 4,0,0,-3,0,0,0,0,0,0
480 REM B
490 DATA 0,0
500 DATA 0,6,3,0,1,-1,0,-1,-1,-1,-3,
510 0,3,0,1,-1,0,-1,-1,-1,-3,0
510 REM C
  
```

```

520 DATA 4,1
530 DATA -1,-1,-2,0,-1,1,0,4,1,1,2,0,
540 1,-1,0,0,0,0,0,0,0,0
540 REM D
550 DATA 0,0
560 DATA 0,6,3,0,1,-1,0,-4,-1,-1,-3,
570 0,0,0,0,0,0,0,0,0,0
570 REM E
580 DATA 4,1
590 DATA -1,-1,-3,0,0,3,3,0,-3,0,0,3,
600 3,0,1,-1,0,0,0,0,0,0
600 REM F
610 DATA 0,0
620 DATA 0,3,3,0,-3,0,0,3,3,0,1,-1,
630 0,0,0,0,0,0,0,0,0,0
630 REM G
640 DATA 2,3
650 DATA 2,0,0,-2,-1,-1,-2,0,-1,1,0,
660 4,1,1,2,0,1,-1,0,0,0,0
660 REM H
670 DATA 0,0
680 DATA 0,6,0,-3,4,0,0,3,0,-6,0,0,0,
690 0,0,0,0,0,0,0,0,0,0
690 REM I
700 DATA 2,0
710 DATA 0,6,0,0,0,0,0,0,0,0,0,0,0,
720 0,0,0,0,0,0,0,0,0,0
720 REM J
730 DATA 0,2
740 DATA 0,-1,1,-1,1,0,1,1,0,5,1,0,-
750 3,0,0,0,0,0,0,0,0,0
750 REM K
760 DATA 0,0
770 DATA 0,6,0,-3,2,0,2,3,-2,-3,2,-3,
780 0,0,0,0,0,0,0,0,0,0
780 REM L
790 DATA 0,6
800 DATA 0,-6,3,0,1,1,0,0,0,0,0,0,0,
810 0,0,0,0,0,0,0,0,0,0
810 REM M
820 DATA 0,0
830 DATA 0,6,2,-3,2,3,0,-6,0,0,0,0,0,
840 0,0,0,0,0,0,0,0,0,0
840 REM N
850 DATA 0,0
860 DATA 0,6,4,-6,0,6,0,0,0,0,0,0,0,
870 0,0,0,0,0,0,0,0,0,0
870 REM O
880 DATA 0,1
890 DATA 0,4,1,1,2,0,1,-1,0,-4,-1,-1,
900 -2,0,-1,1,0,0,0,0,0,0
900 REM P
910 DATA 0,0
920 DATA 0,6,3,0,1,-1,0,-1,-1,-1,-3,
930 0,0,0,0,0,0,0,0,0,0
930 REM Q
940 DATA 2,2
950 DATA 1,-2,-2,0,-1,1,0,4,1,1,2,0,
960 1,-1,0,-4,-1,-1,0,0,0,0
960 REM R
970 DATA 0,0
980 DATA 0,6,3,0,1,-1,0,-1,-1,-1,-3,
990 0,2,0,2,-3,0,0,0,0,0,0
990 REM S
1000 DATA 0,1
1010 DATA 1,-1,2,0,1,1,0,1,-1,1,-2,0,
1020 -1,1,0,1,1,1,2,0,1,-1
1020 REM T
1030 DATA 2,0
1040 DATA 0,6,-2,0,4,0,0,0,0,0,0,0,0,
1050 0,0,0,0,0,0,0,0,0,0
  
```

```

1050 REM U
1060 DATA 0,6
1070 DATA 0,-4,1,-2,2,0,1,2,0,4,0,0,0
      ,0,0,0,0,0,0,0,0,0,0
1080 REM V
1090 DATA 0,6
1100 DATA 0,-3,2,-3,2,3,0,3,0,0,0,0,0
      ,0,0,0,0,0,0,0,0,0,0
1110 REM W
1120 DATA 0,6
1130 DATA 0,-6,2,3,2,-3,0,6,0,0,0,0,0
      ,0,0,0,0,0,0,0,0,0,0
1140 REM X
1150 DATA 0,0
1160 DATA 0,1,4,4,0,1,0,-1,-2,-2,-2,2
      ,0,1,0,-1,4,-4,0,-1,0,0
1170 REM Y
1180 DATA 2,0
1190 DATA 0,3,-2,3,2,-3,2,3,0,0,0,0,0
      ,0,0,0,0,0,0,0,0,0,0
1200 REM Z
1210 DATA 4,1
1220 DATA -1,-1,-3,0,4,6,-3,0,-1,-1,0
      ,0,0,0,0,0,0,0,0,0,0
9000 REM -----
9001 REM LETRAS EN HI-RES
9002 REM -----
9023 REM -----
9100 D = D + 4
9110 IF MOV = 1 THEN SX = 1
      : SY = 1
      : M1 = 1
      : M2 = 2
9120 IF MOV = 2 THEN SX = 1
      : SY = -1
      : M1 = 2
      : M2 = 1
9130 IF MOV = 3 THEN SX = -1
      : SY = -1
      : M1 = 1
      : M2 = 2
9140 IF MOV = 4 THEN SX = -1
      : SY = 1
      : M1 = 2
      : M2 = 1
9150 LU = LEN (A$)
9160 FOR I = 1 TO LU
9170 LE$ = MID$(A$,I,1)
9180 COD = ASC (LE$)
      : NL = COD - 64
9190 IF COD < 65 OR COD > 90 THEN 9260

      : REM A VARIAR SI SE AÑADEN
      : NUEVOS CARACTERES
9200 X1 = X0 + (S * POZ(NL,M1) * SX)
      : Y1 = Y0 - (S * POZ(NL,M2) * SY)
9210 FOR J = 1 TO 11
9220 X2 = X1 + (S * CAR$(NL,J,M1) * SX
      )
      : Y2 = Y1 - (S * CAR$(NL,J,M2) * SY
      )
9230 HPLOT X1,Y1 TO X2,Y2
9240 X1 = X2
      : Y1 = Y2
9250 NEXT
9260 IF MOV = 2 OR MOV = 4 THEN YO
      = YO - (D * S * SY)
      : GOTO 9280
9270 X0 = X0 + (D * S * SX)
9280 NEXT
9290 RETURN

```

En las figuras de las páginas 1495 y 1496 se han representado los diagramas de flujo del programa, y en las páginas 1496 a 1498 el listado. El programa se compone de tres bloques principales. El main (líneas 10,90) contiene la definición de las áreas de memoria (DIM) y las llamadas a las subrutinas. De éstas, la 350 carga con una serie de instrucciones DATA los valores de los desplazamientos, y la 9100 las ejecuta, presentando la escritura. En esta versión, tanto la cadena a presentar A\$ como los parámetros X0, Y0, S, D y MOV son introducidos por teclado. Para generalizar el programa para que pueda utilizarse en todas las aplicaciones, es suficiente con convertirlo en una subrutina y con transferir los valores adquiridos o determinados en otros puntos.

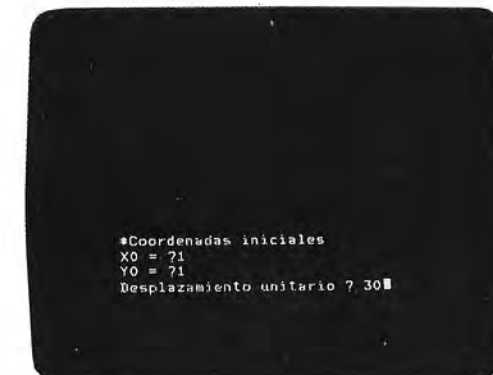
Los parámetros utilizados tienen el siguiente significado:

- X0,Y0 Coordenadas de inicio de escritura, o sea del extremo de referencia del rectángulo que contiene la letra.
- S Escala. Este factor determina las dimensiones la presentación. Multiplicando por S los desplazamientos que determinan un carácter se obtiene el mismo carácter ampliado.
- D Espaciado. Define la distancia entre un carácter y otro.
- MOV Indica la orientación de la escritura. Las orientaciones previstas, codificadas con los números 1 a 4, se indican en el listado.

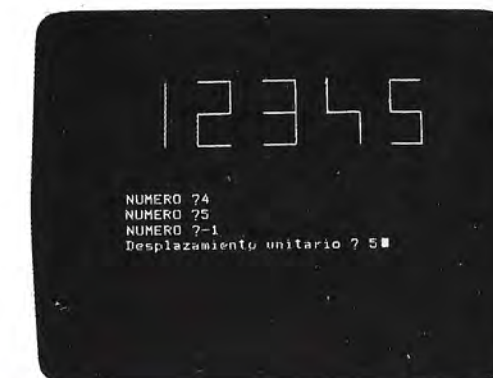
PRESENTACION DE CARACTERES NUMERICOS EN MODO GRAFICO

La secuencia fotográfica ilustra un ejemplo de aplicación del programa de representación de los caracteres numéricos en modo gráfico.

Aquí al lado se ha representado el coloquio inicial con el usuario, fase en la que se produce la introducción de las coordenadas iniciales y del valor del desplazamiento unitario.



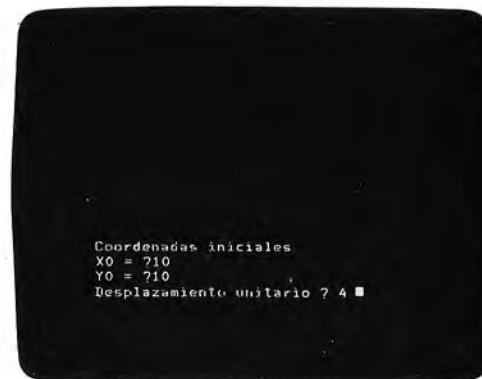
A continuación, el programa activa el modo gráfico, conservando una ventana de cuatro líneas en modalidad de texto para el coloquio. El usuario ha introducido una serie de cifras a representar. Después de cada emisión, el programa presenta las cifras pedidas. La introducción de un número negativo produce la nueva selección del desplazamiento unitario.



Al seleccionar un desplazamiento unitario más pequeño, las dimensiones de las cifras se reducen. Sin embargo, debe observarse que las proporciones siempre son las mismas.



En este caso se han variado los valores de las coordenadas iniciales y del desplazamiento unitario.



Las cifras presentadas con desplazamiento unitario igual a 4 todavía son más grandes que las cifras ASCII estándar, visibles en la parte reservada al coloquio.



La última fotografía muestra la progresiva reducción de escala que se tiene al disminuir el valor del desplazamiento unitario.



PRESENTACION DE CARACTERES ALFABETICOS EN MODO GRAFICO

En esta página pueden verse algunos ejemplos de empleo de los caracteres alfabéticos en modalidad gráfica.

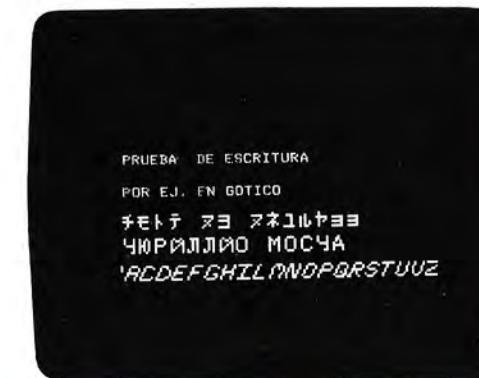
Aquí al lado se ha presentado el resultado de la ejecución del programa presentado. Se ha introducido la cadena BASIC.



La segunda fotografía muestra las diversas posibilidades ofrecidas por el programa. Las leyendas pueden posicionarse en cualquier punto de la pantalla y con la orientación deseada.



Elaborando más profundamente el esquema de trazado de los caracteres es posible reproducir diversos juegos de caracteres tipográficos. La fotografía de aquí al lado muestra algunas prestaciones de un paquete software dedicado.



Histogramas y diagramas de tarta

La representación gráfica de un fenómeno cualquiera puede obtenerse trazando el proceso de la función matemática que lo describe, siempre que exista. Hay fenómenos que no pueden ser descritos mediante una función por el sencillo motivo de que la cantidad a representar no depende de una variable (independiente) que varía en continuidad, o bien porque la variable independiente no es de tipo numérico.

En su momento se expuso un ejemplo relativo a las ventas de un artículo por parte de varios vendedores; en aquella ocasión no era posible considerar a los vendedores (A, B, C,...) como datos numéricos. Otro ejemplo muy corriente se encuentra en la representación del balance mensual a lo largo de un período anual. En este caso, la «variable independiente» es el nombre del mes (enero, febrero,...), que no es una cantidad numérica. Para resolver estos problemas existen dos métodos muy difundidos: los histogramas y los diagramas de tarta.

Los histogramas se presentan como una serie

Aplicación gráfica en el control de proceso.



K. Reeser/Marka

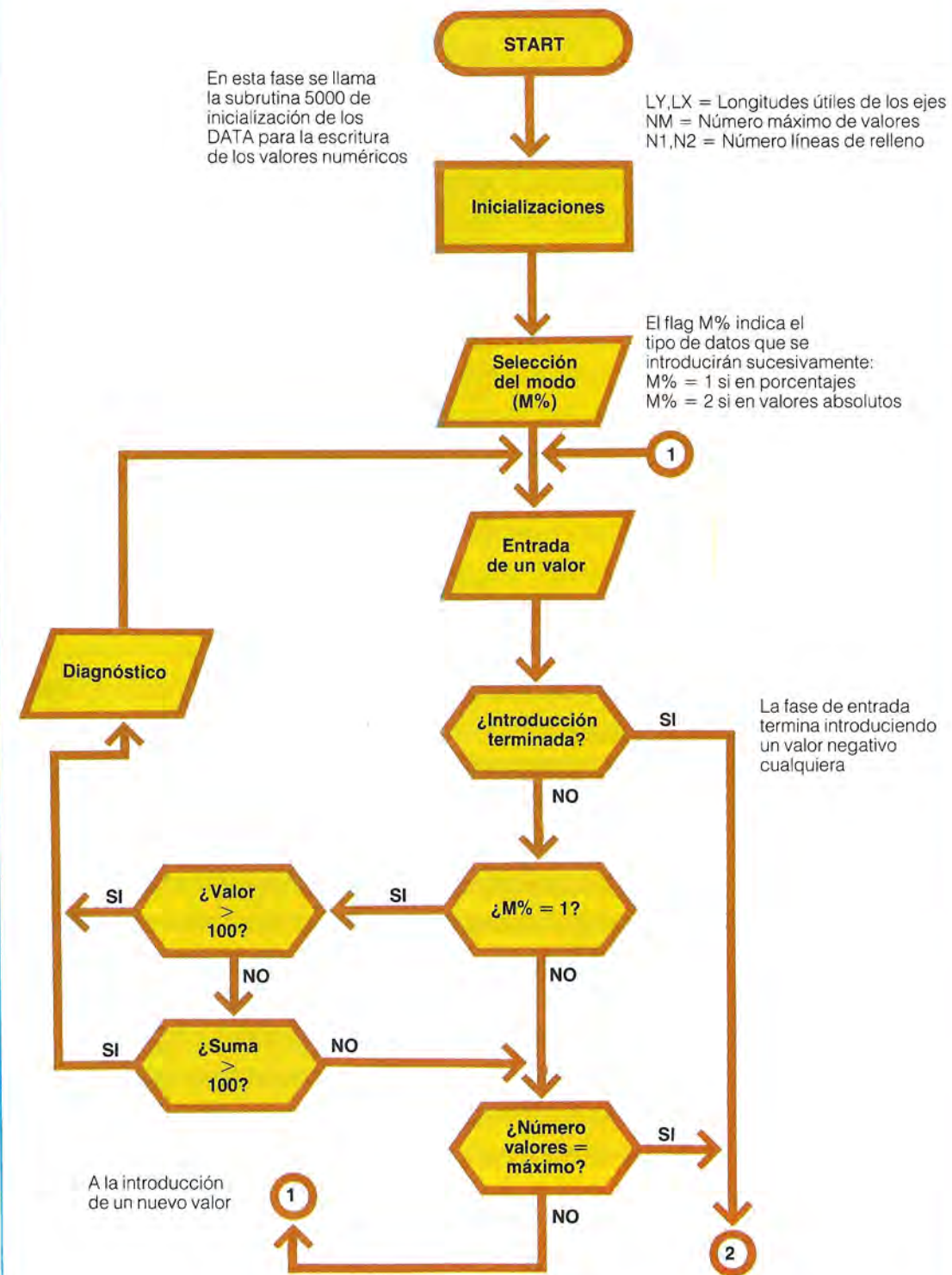
de columnas de altura proporcional a los valores a representar según el eje Y y distanciados sobre el eje X en cantidades proporcionales a los valores de la variable independiente (X). Generalmente son de paso constante, puesto que el fenómeno se observa a intervalos regulares. En el caso del balance, la «variable independiente» es el tiempo y la dependiente el importe; los valores se indican con una periodicidad mensual, por lo que el eje X está dividido en 12 partes, mientras que el eje Y puede representar la cifra mensual tanto en valor absoluto como en un valor porcentual del total. Esta última forma es la más empleada, puesto que proporciona una visión más inmediata del fenómeno y facilita la construcción del diagrama.

Los diagramas de tarta son representaciones similares a los histogramas, pero cada valor a indicar es un sector circular de amplitud proporcional. La circunferencia completa representa el total de los valores, y cada sector en que está dividida representa la entidad de cada componente como porcentaje del total. Por ejemplo, para representar el anterior programa de balance de esta manera, debe construirse un círculo dividido en 12 sectores, uno por mes, de amplitud proporcional al importe de cada mes.

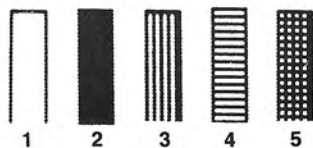
El histograma proporciona una visión del proceso del balance en el tiempo, mientras que el diagrama de tarta muestra cómo se distribuye el balance entre las varias divisiones (meses). Por tanto, se trata de dos representaciones que se complementan, o sea no son alternativas.

En las figuras de las páginas 1503 y 1504 se ha representado el diagrama de flujo de un programa para la representación de histogramas (el listado puede verse en las págs. 1504 a 1510). Los datos a representar pueden ser proporcionados tanto como porcentaje como en valor absoluto, y en este último caso es el programa el que procede a convertirlos en porcentajes. Además, el usuario puede elegir 5 modos de representación diferentes; de esta manera es posible trazar hasta cinco histogramas distintos sobre el mismo gráfico, distinguiéndose con facilidad. En el programa, los cinco modos de representación están proyectados haciendo referencia a una pantalla monocromática. Esto complica la subrutina de presentación, puesto que debe prever varios modos de representación de las columnas del histograma. En el vídeo en colores, a cada modo de representación puede asociarse un color diferente.

REPRESENTACION GRAFICA DE HISTOGRAMAS



- 1 = Vacío
- 2 = Lleno
- 3 = Líneas verticales
- 4 = Líneas horizontales
- 5 = Entrecruzado



Modo de presentación



REPRESENTACION DE HISTOGRAMAS

Versión Siprel, Apple y compatibles

```

4 REM -----
5 REM HISTOGRAMAS
6 REM -----
7
8
9 DIM PD%(26,2),CAR%(26,11,2)
10 LX = 230
   LY = 150
20 NM = 20
30 P% = 0
50 DIM D(NM),P(NM),S(NM),E$(NM)
60 GOSUB 5000
65 GOSUB 9120
70 TEXT
   : HOME
80 VTAB 12
   : INPUT "LOS DATOS SON EN PORCENTAJE
   ? (S/N) ";A$
90 IF A$ = "S" THEN M% = 1
   : P% = 1
   : GOTO 110
100 M% = 2
110 I = 1
   : TT = 0
120 HOME

```

```

   : VTAB 12
130 PRINT "INTRODUZCA EL DATO N. "1
140 INPUT "(NEGATIVO PARA TERMINAR) ";D
   $
   : IF D$ = "" THEN 120
150 D = VAL (D$)
160 IF D > 99999 THEN 120
170 IF D < 0 THEN 260
180 D(I) = INT (D)
190 TT = TT + D(I)
200 IF M% = 2 THEN 220
210 IF D(I) > 100 OR TT > 100 THEN
   GOSUB 6000
   : HOME
   : GOTO 110
   : REM ERROR
220 VTAB 15
   : INPUT "ETIQUETA (MAX. 3 CARACTERES
   ): ";E$
230 E$(I) = LEFT$(E$,3)
240 I = I + 1
   : IF I > NM THEN 260
250 GOTO 120
260 IF TT = 0 THEN RUN
270 NV = I - 1
280 HOME
290 VTAB 5
   : PRINT "MODOS DE PRESENTACION:"
300 VTAB 10
310 PRINT "1) VACIO"
320 PRINT "2) LLENO"
330 PRINT "3) LINEAS VERTICALES"
340 PRINT "4) LINEAS HORIZONTALES"
350 PRINT "5) ENTRECruzADO"
360 VTAB 22
   : INPUT "ELIJA: ";MP$
   : IF MP$ = "" THEN 360
370 MP = VAL (MP$)
380 IF MP < 1 OR MP > 5 THEN 280
1000 REM -----
1010 REM PROCESO
1020 REM -----
1030 IF M% = 1 GOTO 1140
1040 HOME
   : VTAB 12
1050 PRINT "QUIERE LA REPRESENTACION
   EN PORCENTAJE?";
1060 INPUT "(S/N) ";R$
1070 IF R$ = "S" THEN P% = 1
   : GOTO 1140
1080 MA = D(1)
1090 FOR K = 2 TO NV
1100 IF D(K) > (MA) THEN MA = D(K)
1120 NEXT K
1130 TT = MA
1140 FOR I = 1 TO NV
1150 P(I) = D(I) / TT * 100
1160 S(I) = INT (LY * P(I) / 100)
1170 NEXT I
1180 LA = INT (LX / (2 * NV))
2000 REM -----
2001 REM PARAMETROS PRESENTACION
2002 REM -----
2010: SX = 1
   : SY = 1
2040 ON MP GOTO 3000,3000,2060,2070,20
   80
2060 SX = 2
   : GOTO 3000
2070 SY = 2
   : GOTO 3000

```

```

2080 SY = 2
: SX = 2
3000 REM -----
3010 REM DIBUJA VENTANA
3020 REM -----
3030 HGR2
: HCOLOR= 3
3040 HPLLOT 34,0 TO 279, 0 TO 279, 159
: TO 34,159 TO 34,0
3050 FOR I = 1 TO 10
3060 V = 159 - I * 15
3070 HPLLOT 34,V TO 39,V
3075 HPLLOT 274,V TO 278,V
3080 NEXT
3090 S = 3
: YO = 159
3100 IF P% = 0 THEN 3170
3110 FOR NN = 0 TO 100 STEP 10
3120 XO = 9 * S
3130 GOSUB 5500
3140 YO = YO - 15
3150 NEXT
3160 GOTO 3250
3170 ST = MA / 10
: IF ST < > INT (ST) THEN ST =
: INT (ST) + 1
3175 YO = 9
3180 FOR NN = MA TO 0 STEP - ST
3190 XO = 9 * 5 + 3
3200 GOSUB 5500
3210 YO = YO + 15
3220 NEXT
3230 XO = 9 * 5 + 3
3235 IF NN + ST = 0 THEN 3250
3240 NN = 0
: GOSUB 5500
3250 REM
4000 REM -----
4001 REM DIBUJA HISTOGRAMAS
4002 REM -----
4005 N = 0
4007 Y1 = 159
4010 FOR I = 1 TO NV
4025 X1 = 50 + N * LA
4027 X2 = X1 + LA
4030 HPLLOT X1,Y1 TO X1,Y1 - S(I) TO X2
: ,Y1 - S(I) TO X2,Y1
4040 IF MP = 1 THEN 4900
4050 IF MP = 4 THEN 4300
4060 Y = Y1
: X = X1 + SX
4070 IF X > X2 THEN 4200
4080 HPLLOT X,Y TO X,Y - S(I)
4090 X = X + SX
4100 GOTO 4070
4200 IF MP = 5 THEN 4300
4210 GOTO 4900
4300 X = X1
: Y = Y1 - SY
4310 IF Y < Y1 - S(I) THEN 4900
4320 HPLLOT X1,Y TO X2,Y
4330 Y = Y - SY
4340 GOTO 4310
4900 N = N + 2
4905 A# = E$(I)
: HO = X2
: VO = 181
: S = 1
: D = 2
: MOV = 4
: GOSUB 10270

```

```

4910 NEXT I
4920 GET F#
4930 TEXT
: HOME
: VTAB 12
: INPUT "QUITERE OTRO GRAFICO? ";R
$
4940 IF LEFT$(R#,1) = "S" THEN 70
4950 END
5000 REM -----
5010 REM INICIALIZA NUMEROS
5020 REM -----
5030 DIM XD(7),YD(7),XA(7),YA(7)
5040 DIM V(7),CX(10,7)
5050 FOR I = 1 TO 7
5060 READ XD(I),YD(I),XA(I),YA(I)
5070 NEXT I
5080 FOR I = 1 TO 10
5090 FOR J = 7 TO 1 STEP - 1
5100 READ CX(I,J)
: NEXT J
5110 NEXT I
5120 RETURN
5500 REM -----
5510 REM DIBUJA NUMEROS
5520 REM -----
5530 NN# = STR$(NN)
5540 LN = LEN (NN#)
5550 FOR KK = LN TO 1 STEP -1
5560 N = VAL < MID$(NN#,KK,1)
5570 XO = XO - 2 * 5
5580 FOR J = 1 TO 7
: V(I) = 0
: NEXT I
5590 LX = N
: IF LX = 0 THEN LX = 10
5600 FOR I = 1 TO 7
5610 V(I) = CX(LX,I)
5620 NEXT I
5630 FOR I = 1 TO 7
5640 IF V(I) = 0 GOTO 5710
5650 K = I
5660 X1 = XO + XD(K) * S
5670 Y1 = YO - YD(K) * S
5680 X2 = XO - XA(K) * S
5690 Y2 = YO - YA(K) * S
5700 HPLLOT X1,Y1 TO X2,Y2
5710 NEXT I
5720 NEXT KK
5730 RETURN
6000 VTAB 20
: PRINT "INTRODUCCION ERRONEA:"
6010 PRINT
: PRINT "LA SUMA DATOS SUPERA EL VA
: LOR '100'"
6015 PRINT
6020 PRINT "PULSAR UN TECLA PARA VOLVER
: A EMPEZAR >";
6030 GET Q#
6040 RETURN
7945 REM -----
7950 REM DATA
7955 REM -----
8000 DATA 0,0,0,1
: REM =a
8010 DATA 0,1,0,2
: REM =b
8020 DATA 0,2,1,2
: REM =c
8030 DATA 1,2,1,1
: REM =d

```

```

8040 DATA 1,1,1,0
: REM =e
8050 DATA 1,0,0,0
: REM =f
8060 DATA 0,1,1,1
: REM =g
8070 DATA 0,0,1,1,0,0,0
: REM =1
8080 DATA 1,1,0,1,1,0,1
: REM =2
8090 DATA 1,1,1,1,1,0,0
: REM =3
8100 DATA 1,0,1,0,0,1,0
: REM =4
8110 DATA 1,1,1,0,1,1,0
: REM =5
8120 DATA 1,1,1,0,0,1,1
: REM =6
8130 DATA 0,0,1,1,1,0,0
: REM =7
8140 DATA 1,1,1,1,1,1,1
: REM =8
8150 DATA 1,0,1,1,1,1,0
: REM =9
8160 DATA 0,1,1,1,1,1,1
: REM =0
9090 REM -----
9100 REM INICIALIZACION
9110 REM -----
9120 FOR I = 1 TO 26
9130 READ POX(I,1),POX(I,2)
9140 FOR J = 1 TO 11
9150 READ CARX(I,J,1),CARX(I,J,2)
9160 NEXT J
9170 NEXT I
9180 RETURN
9190 REM -----
9200 REM DATA
9210 REM -----
9220 REM A
9230 DATA 0,0
9240 DATA 0,4,1,2,2,0,1,-2,0,-1,-4,0,
4,0,0,-3,0,0,0,0,0,0
9250 REM B
9260 DATA 0,0
9270 DATA 0,6,3,0,1,-1,0,-1,-1,-1,-3,
0,3,0,1,-1,0,-1,-1,-1,-3,0
9280 REM C
9290 DATA 4,1
9300 DATA -1,-1,-2,0,-1,1,0,4,1,1,2,0,
1,-1,0,0,0,0,0,0,0,0
9310 REM D
9320 DATA 0,0
9330 DATA 0,6,3,0,1,-1,0,-4,-1,-1,-3,
0,0,0,0,0,0,0,0,0,0
9340 REM E
9350 DATA 4,1
9360 DATA -1,-1,-3,0,0,3,3,0,-3,0,0,3,
3,0,1,-1,0,0,0,0,0,0
9370 REM F
9380 DATA 0,0
9390 DATA 0,3,3,0,-3,0,0,3,3,,0,1,-1,
0,0,0,0,0,0,0,0,0,0
9400 REM G
9410 DATA 2,3
9420 DATA 2,0,0,-2,-1,-1,-2,0,-1,1,0,
4,1,1,2,0,1,-1,0,0,0,0
9430 REM H
9440 DATA 0,0
9450 DATA 0,6,0,-3,4,0,0,3,0,-6,0,0,0,
0,0,0,0,0,0,0,0,0,0

```

```

9460 REM I
9470 DATA 2,0
9480 DATA 0,6,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
9490 REM J
9500 DATA 0,2
9510 DATA 0,-1,1,-1,1,0,1,1,0,5,1,0,-
3,0,0,0,0,0,0,0,0,0
9520 REM K
9530 DATA 0,0
9540 DATA 0,6,0,-3,2,0,2,3,-2,-3,2,-3,
0,0,0,0,0,0,0,0,0,0
9550 REM L
9560 DATA 0,6
9570 DATA 0,-6,3,0,1,1,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
9580 REM M
9590 DATA 0,0
9600 DATA 0,6,2,-3,2,3,0,-6,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
9610 REM N
9620 DATA 0,0
9630 DATA 0,6,4,-6,0,6,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
9640 REM O
9650 DATA 0,1
9660 DATA 0,4,1,1,2,0,1,-1,0,-4,-1,-1,
-2,0,-1,1,0,0,0,0,0,0
9670 REM P
9680 DATA 0,0
9690 DATA 0,6,3,0,1,-1,0,-1,-1,-1,-3,
0,0,0,0,0,0,0,0,0,0
9700 REM Q
9710 DATA 2,2
9720 DATA 1,-2,-2,0,-1,1,0,4,1,1,2,0,
1,-1,0,-4,-1,-1,0,0,0,0
9730 REM R
9740 DATA 0,0
9750 DATA 0,6,3,0,1,-1,0,-1,-1,-1,-3,
0,2,0,2,-3,0,0,0,0,0,0
9760 REM S
9770 DATA 0,1
9780 DATA 1,-1,2,0,1,1,0,1,-1,1,-2,0,
-1,1,0,1,1,1,2,0,1,-1
9790 REM T
9800 DATA 2,0
9810 DATA 0,6,-2,0,4,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
9820 REM U
9830 DATA 0,6
9840 DATA 0,-4,1,-2,2,0,1,2,0,4,0,0,0,
0,0,0,0,0,0,0,0,0,0
9850 REM V
9860 DATA 0,6
9870 DATA 0,-3,2,-3,2,3,0,3,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
9880 REM W
9890 DATA 0,6
9900 DATA 0,-6,2,3,2,-3,0,6,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
9910 REM X
9920 DATA 0,0
9930 DATA 0,1,4,4,0,1,0,-1,-2,-2,-2,2,
0,1,0,-1,4,-4,0,-1,0,0
9940 REM Y
9950 DATA 2,0
9960 DATA 0,3,-2,3,2,-3,2,3,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0
9970 REM Z
9980 DATA 4,1
9990 DATA -1,-1,-3,0,4,6,-3,0,-1,-1,0,
0,0,0,0,0,0,0,0,0,0

```

```

10000 REM -----
10010 REM      LETRAS EN HI-RES
10020 REM -----
10030 REM
10040 REM
10050 REM      EN ENTRADA:
10060 REM
10070 REM      A# - CADENA DE CA-
10080 REM            RACTERES
10090 REM      H0,V0 - POSICION DESDE
10100 REM            DONDE EMPEZAR
10110 REM            A ESCRIBIR
10120 REM      S - ESCALA (>0)
10130 REM      D - ESPACIADO (>0)
10140 REM      MOV - TIPO DE ESCRI-
10150 REM            TURA:
10160 REM
10170 REM            1 = HORIZONTAL
10180 REM            2 = DESCENDENTE
10190 REM            3 = AL REVES
10200 REM            4 = ASCENDENTE
10210 REM
10220 REM
10230 REM -----
10270 D = D + 4
10280 IF MOV = 1 THEN PX = 1
      : PY = 1
      : M1 = 1
      : M2 = 2
10290 IF MOV = 2 THEN PX = 1
      : PY = -1
      : M1 = 2
      : M2 = 1
10300 IF MOV = 3 THEN PX = -1
      : PY = -1
      : M1 = 1
      : M2 = 2
10310 IF MOV = 4 THEN PX = -1
      : PY = 1
      : M1 = 2
      : M2 = 1
10320 LU = LEN (A#)
10330 FOR II = 1 TO LU
10340 LE# = MID# (A#,II,1)
10345 IF LE# = " " THEN II = LU
      : NEXT II
      : RETURN
10350 COD = ASC (LE#)
      : NL = COL - 64
10360 IF COD < 65 OR COD > 90 THEN 1043
      0
      : REM A VARIAR SI SE AÑADEN
      NUEVOS CARACTERES
10370 H1 = H0 + (S * POX(NL,M1) * PX)
      : V1 = V0 - (S * POX(NL,M2) * PY)
10380 FOR J = 1 TO 11
10390 H2 = H1 + (S * CARX(NL,J,M1) * PX
      )
      : V2 = V1 - (S * CARX(NL,J,M2) * PY
      )
10400 HPL0T H1,V1 TO H2,V2
10410 H1 = H2
      : V1 = V2
10420 NEXT J
10430 IF MOV = 2 OR MOV = 4 THEN V0
      = V0 - (D * S * PY)
      : GOTO 10450
10440 H0 = H0 + (D * S * PX)
10450 NEXT II
10460 RETURN

```

El ordenador y los hombres radar

Seguramente todo el mundo ha oído hablar de los controladores de tráfico aéreo, los llamados «hombres radar», pero pocas personas conocen verdaderamente cuáles son sus actividades y cómo el soporte de las nuevas tecnologías se ha convertido en condición irrenunciable para el buen desarrollo de su trabajo.

Las condiciones de funcionamiento de un centro de control sin ninguna automatización (control de procedimiento) pueden automatizarse gradualmente. Es importante poner de manifiesto esta gradualidad, porque hoy, en el mundo hay ejemplos que cubren todo el espectro de los posibles grados de automatización, desde el control de procedimiento hasta el intercambio de datos entre ordenadores.

La elección que cada nación hace está claramente ligada al costo de la automatización y a los beneficios que ésta comporta. En el caso de la vigilancia radar, en el costo entran los costos y el mantenimiento de las instalaciones radar y de los sistemas automáticos, el adiestramiento de personal altamente cualificado y el eventual grado de dependencia de los países productores de tecnología, mientras que por beneficios debe entenderse la mayor seguridad y la mejor cualificación del trabajo.

El control del tráfico aéreo (ATC, Air Traffic Control) es un servicio cuyo fin es asegurar un tráfico seguro, ordenado y rápido en interés de los viajeros y de las compañías aéreas en los espacios aéreos de todo el mundo. La organización mundial que supervisa y coordina las complejas problemáticas del ATC es la ICAO (Internacional Civil Aviation Organization), a la que están adheridos todos los países del mundo.

Para hacer más ágil el control del tráfico, el espacio aéreo mundial está dividido en FIR (Flight Information Region), cuyo control está adjudicado a las naciones geográficamente interesadas. Para simplificar las dificultades de coordinación del tráfico, en el cielo se han trazado idealmente las aerovías, de unas 10 millas de anchura, y divididas en varios niveles de distintas cotas, a lo largo de las cuales vuelan los aviones.

Contrariamente a lo que podría pensarse, el servicio de control se extiende a toda la duración del vuelo, y no sólo a la fase inicial (partida) y final (aproximación a la pista y aterrizaje).

La misión del controlador es guiar vía radio a los pilotos a lo largo de las aerovías, considerando



Las tres regiones de vuelo (FIR) en que está subdividido el espacio aéreo italiano.

las peticiones de los propios pilotos y la situación del tráfico en cada momento, que sólo el controlador conoce en su totalidad, puesto que la visual del piloto es extremadamente limitada, y también a causa de su velocidad (un avión de línea recorre unos 10 km por minuto).

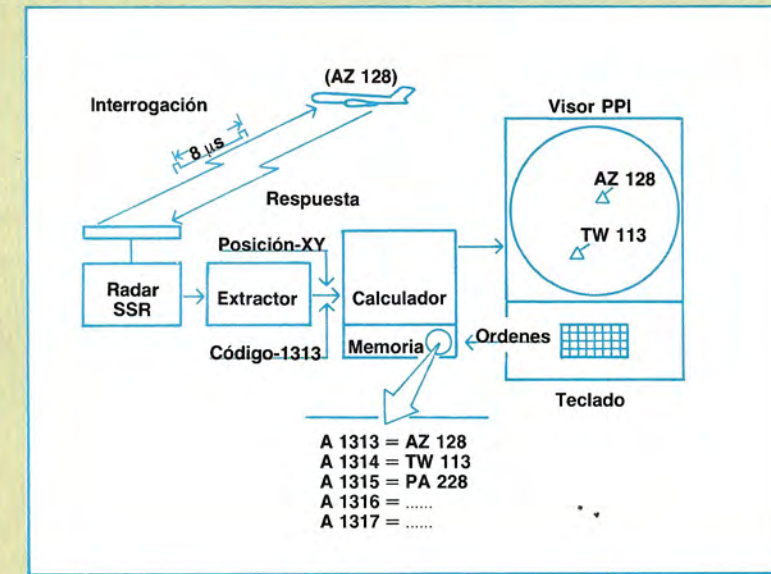
Para agilizar la misión del controlador, antes de la partida, los pilotos están obligados a redactar el plan de vuelo (FPL, Flight Plane), que contiene informaciones sobre la denominación del avión, sobre sus características técnicas (tipo) y sobre la ruta del aeropuerto de partida y el de llegada, completadas con los horarios previstos. En el plan de vuelo, el itinerario está expresado como sucesiones de aerovías o de puntos característicos (FIX), que muy a menudo corresponden a radiofaros de tierra, presentes en las cartas internacionales.

El plan de vuelo se transmite a todas las regiones de vuelo (FIR) que hay en la ruta mediante una red telex dedicada llamada AFTN (Aero-

nautical Fixed Telecommunication Network). En cada región de vuelo, los asistentes de control tienen la misión de extraer de cada plan de vuelo las informaciones de ruta que corresponden a su región específica. Para cada FIX interesado por la ruta debe producirse una tira de papel (strip) en la que se indican los datos esenciales del avión, la cota pedida para el vuelo, el nombre del FIX y el horario de sobrevuelo previsto. Para poner un ejemplo, la FIR de Roma prepara de 10 a 15 strips por vuelo, y con una base de unos 900 vuelos diarios, se llega a unas 10.000 tiras al día. El sistema que permite al controlador conocer en cada instante el tráfico correspondiente a un FIX, consiste en ordenar cronológicamente las strips que corresponden al punto en cuestión para comprobar que no haya previstos cruces de varios aviones en el mismo nivel sin la debida distancia. Para un control de este tipo (llamado de procedimiento), la separación requerida entre dos aviones es típicamente de 10 minutos si vuelan en la misma cota, o bien de unos 330 m (1.000 pies) si vuelan en cotas diferentes. Un momento operativamente muy delicado es aquel en que un avión atraviesa el confín entre dos regiones de vuelo. Una red telefónica dedicada a este objeto permite a los controladores de dos regiones limítrofes coordinar conjuntamente el punto, la cota y el momento de transferencia del control. Mediante esta comunicación,

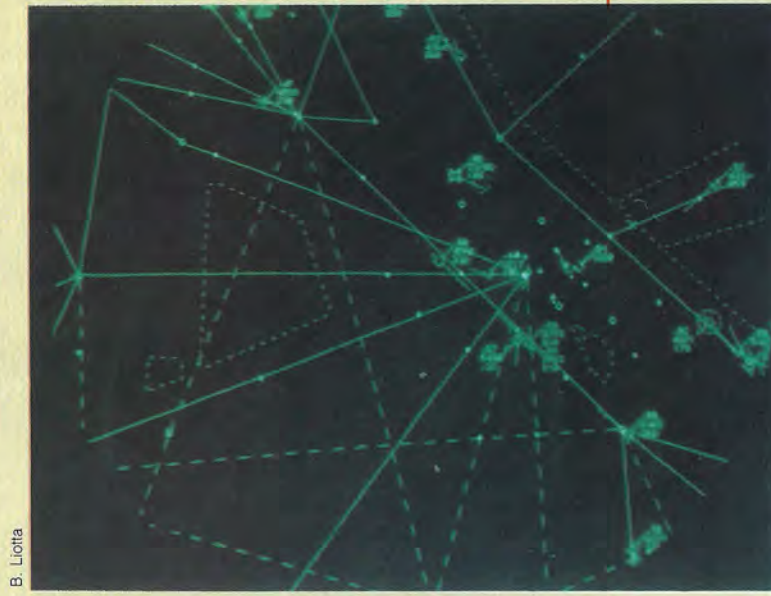
el nuevo controlador puede evaluar qué horarios indicados en las strips deben corregirse para sincronizar la trayectoria prevista a la posición real del avión. Unas comunicaciones sucesivas con el piloto en el momento de sobrevolar varios FIX permiten evaluar eventuales retrasos o adelantos del avión con respecto al horario previsto. La primera aplicación tecnológica de detección en el ATC, realizada a caballo de los años 40-50, fue el empleo del radar, que permitió la presentación en pantalla del tráfico en cada momento, a pesar de que las imágenes generadas sólo eran una serie de puntos luminosos sobre un monitor (los ecos radar). Entre 1958 y 1960 se introdujo el uso de un segundo radar, llamado SSR (Secondary Surveillance Radar), que interroga un aparato retransmisor denominado transponder que va a bordo del aparato. La respuesta digital que obtiene el SSR del transponder contiene una serie de informaciones que corresponden a la identidad del avión (expresada en un código de 16 bits) y la cota a la que vuela. Estas comunicaciones digitales permiten el uso del procesador para la identificación de los vuelos en el monitor PPI (Plan Position Indicator). El primer grado de automatización es el llamado labelling, que consiste en asociar a la representación del eco del radar primario una etiqueta que indica el código del avión, su cota y otros datos útiles a las actividades de control.

Después de un primer contacto por radio con el piloto, el controlador puede sustituir el código de identificación que el ordenador recibe del transponder por el nombre efectivo del vuelo (por ejemplo, AZ 128), que permitirá una confrontación más fácil entre la posición del avión, deducida ahora de las informaciones radar, y las strips (gráfico de al lado). El procesador permite asociar a la presentación en el monitor PPI otras numerosas funciones útiles para el control, como por ejemplo la presentación de una vista esquemática de las aerovías, de los FIX y de las zonas particulares que hay en una región de vuelo, como las zonas de aeropuertos o las excluidas al tráfico civil, la presentación, si se desea, de las cotas de los principales relieves próximos al aeropuerto, las cartas de los itinerarios de aproximación a las pistas, el cálculo de la distancia entre dos aviones señalados y el cambio de escala en la representación en el monitor. Con el empleo del seguimiento se obtiene una fase ulterior de automatización, que se realiza mediante el proceso de las informaciones radar. A través del responder, el avión sólo da informaciones relativas a la cota, mientras que los ecos de respuesta de los radares primario y secundario dan informaciones correspondientes a la distancia. El seguimiento es la función de correlación de estos datos en el tiempo para trazar un vector de velocidad (intensidad, dirección y sentido) que después se representa en el PPI mediante un símbolo (vector) asociado al que identifica el avión en el vídeo (foto de al lado). El proyecto de la función de seguimiento reviste un notable grado de dificultad, puesto que se obtiene por interpolación de datos radar, inevitablemente afectados de errores. También intervienen ulteriores complicaciones de hardware y de software para las funciones de seguimiento en el caso de que el mismo espacio aéreo esté cubierto por varios radares desplazados a puntos lejanos. Calculadas con precisión la posición y la velocidad de un avión, los datos de salida, además de utilizarse inmediatamente para la presentación en el PPI, pueden constituir la entrada de una función que reviste un interés particular, denominada análisis de los conflictos. Conociendo los datos cinemáticos de un avión, pueden efectuarse algunas previsiones sobre su trayectoria futura y, por tanto, sobre la posibilidad de que pueda ocupar a continuación un



Ejemplo de strips correspondientes a tres puntos de referencia diferentes en el suelo.

ELB			AIE			PNZ		
ELB 12 30	AZ 166 300	LIMM A1	AIE 12 45	AZ 166 300	LIR F	PNZ 13 00	AZ 166 300	
ELB 12 20	TW 200 330	A1	AIE 12 20	AF 270 260		PNZ 12 35	AF 270 260	
ELB 12 15	AZ 334 300		AIE 12 05	TW 200 330		PNZ		
ELB 12 10	AF 270 260		AIE 11 50	LUPO 4 180	LIRA LIRA			



Identificación de un avión a través del responder y del seguimiento vídeo.

determinado punto junto con otro avión. Evidentemente, se trata de una situación de peligro que generará una alarma y la consiguiente y oportuna intervención del controlador. La fase de automatización descrita hasta ahora, seguramente es la más importante por la ayuda que ha aportado a los controladores y el salto de calidad en el plano de la seguridad y de la eficacia. Sometidos al control del tipo descrito, los aviones vuelan con distancias temporales de sólo tres minutos, permitiendo en la fase de ate-

ETO	ELS	GTLE	FLC	FLC	CPL	DIR	ETO	G/S	SR	NE	BEST
-AZ424											4511 04
-AZ957											4507 02
1530 -IG780		218	300	+CHP	32	400	4501	00			
1537 -BA082			300	+KONE	30		4512	05			
1525 -OA134			330	+KONE	15		4514	07			
1522 -IG289			330	+SELS	15		4515	06			
1500 -AZ339			330	+GIRA	15		4510	05			
1517 -AF0644			330	+GIRA	12		4500	04			
1515 -AF0607			330	+GAPC	10		4500	01			

Arriba, presentación en el monitor de las strips correspondientes a varios FIX (puntos de referencia en el suelo). Abajo, un controlador de vuelo en la consola del sistema Satcas de Roma - Ciampino.



rrizaje el despacho del tráfico excesivo y la abolición de los tiempos de espera en cota (holding), que comportaban gravosos incrementos de los costos para las compañías (el coste operativo de un B747 es del orden de medio millón de pesetas por hora).

Una vez resueltos los problemas inherentes a la detección y a la presentación de los aviones como ayuda del ATC, también pueden automatizarse otros aspectos del servicio, con el fin de reducir el trabajo manual y repetitivo de los ayudantes de control.

La interpretación de los planes de vuelo y la impresión mecanizada de las strips pueden considerarse una segunda etapa en el camino de la automatización del ATC.

Para obtener resultados similares, los sistemas han incrementado su complejidad enriqueciéndose con funciones y bases de datos útiles para la decodificación de las informaciones.

Una de las principales bases de datos es aquella en que se describe completamente la geografía en que se desarrolla el tráfico aéreo, mediante la memorización de un conjunto de informaciones interrelacionadas lógicamente, como

por ejemplo todas las aerovías de una región de vuelo, todos los FIX que la constituyen, expresados como nominativo y coordenadas, todos los recorridos de aproximación a las pistas de los diferentes aeropuertos y los respectivos recorridos de partida, los parámetros de las propias pistas, todas las características que corresponden al trayecto de los segmentos de ruta, la relación de las regiones de vuelo limítrofes y los respectivos FIX de los confines.

Todas las informaciones geográficas deben poderse modificar fácilmente, puesto que los nominativos y los puntos notables pueden variar también, o bien pueden abrirse al tráfico nuevas aerovías o anularse otras.

Otra base de datos necesaria para una automatización más eficaz es el archivo de los vuelos de línea con ruta y horarios fijados para un período de seis en seis meses (RPL, Repetitive Plane), que constituyen el 50% de los vuelos diarios a controlar.

Esta base de datos se explora periódicamente (típicamente cada hora) para seleccionar los vuelos cuya entrada en la región de vuelo es inminente. Este archivo también debe ser de fácil gestión, puesto que las compañías proporcionan un nuevo calendario cada seis meses. La mitad restante de los vuelos comunica el propio plan de vuelo a través de la red AFNT. En esta fase de automatización, ésta está conectada directamente al ordenador, que lee el télex y, gracias a unas complejas funciones, realiza una interpretación con la eventual corrección automática de los errores recurrentes. Los mensajes no corregibles se presentan en el vídeo a un operador, que puede aportar las oportunas modificaciones (foto de la pág. 1517).

Los planes repetitivos y los de vuelo recibidos a través de la red AFTN constituyen la entrada para una segunda función automática: la reconstrucción de la ruta.

Apoyándose en datos geográficos, ésta selecciona, de toda la ruta indicada en el plan de vuelo, la parte que interesa a la región de vuelo específica, y la traduce en la sucesión de los FIX interesados consecutivamente en la ruta.

Utilizando algunas bases de datos menores (velocidades y características estándar de los aviones) y las indicaciones horarias indicadas en los planes de vuelo, es posible atribuir automáticamente a cada FIX el correspondiente tiempo de vuelo previsto.

La salida de esta función constituye a su vez la



Sala de operaciones del sistema Satcas de Roma - Ciampino.

entrada de la función final que controla las impresoras para producir las strips durante la jornada. Una tercera fase de automatización consiste en integrar los dos niveles anteriores en un sistema único, que abarca tanto las informaciones del radar como las de las rutas reconstruidas del plan de vuelo. Este segundo paso sólo es posible a condición de disponer de ordenadores de grandes dimensiones, y sobre todo con elevadas velocidades de cálculo.

Técnicamente, la integración se obtiene memorizando la ruta reconstruida en adecuadas bases de datos en las que se ha registrado la evolución del vuelo.

Las informaciones significativas, así como los tiempos de sobrevuelo previstos y reales sobre los FIX y las previsiones de ruta futura, se presentan a los controladores sobre vídeos adecuados, denominados EDD (Electronic Data Display), situados a un lado de la pantalla radar para completar las imágenes del PPI (ver gráfico de la pág. 1517). Conectando después la base de datos radar con la de las rutas reconstruidas es posible comparar la posición efectiva del avión con la ruta declarada para poder generar alarmas en el caso de que se observasen incongruencias.

Las entradas del seguimiento, como la velocidad, pueden ser representadas después en forma digital en los EDD, al lado de los datos del avión extraídos del plan de vuelo, para presen-

tar de manera compacta el máximo de informaciones (foto de la página anterior).

Esta tercera fase de automatización abre un enorme campo de aplicaciones, sobre todo si se proporciona a los controladores una gama exhaustiva de comandos con los que interrogar al sistema sobre la situación del tráfico actual o sobre las previsiones.

En los niveles de automatización más avanzados, el controlador puede modificar las bases de datos mediante oportunos comandos, para informar al sistema de eventuales alteraciones de los planes de vuelo y pedir el recálculo de determinadas rutas y la consiguiente reimpresión de las strips.

Con el paso de los años, con la disponibilidad cada vez mayor de ordenadores con capacidades de cálculo siempre más elevadas y con el conocimiento de que automatizando las funciones de control se obtiene una mayor seguridad, el número de países que han optado por los sistemas ATC automatizados ha aumentado. Actualmente, en Europa son pocas las naciones que no disponen de un servicio similar, y esta situación permite prever un nuevo sistema de automatización: el intercambio de informaciones entre los ordenadores dedicados al control del tráfico aéreo en los diversos países.

La ICAO ya ha previsto protocolos estándar para el intercambio de mensajes entre calculadores, pero todavía falta un acuerdo europeo para llegar a la constitución de una red de ordenadores en la que sea posible intercambiar todas las informaciones sobre los vuelos en tránsito, sustituyendo así la antigua red AFTN. Un enlace similar podría ampliar el concepto de control de flujo a todo el tráfico europeo, coordinando si fuese necesario los cambios de ruta sobre naciones diferentes a las previstas, y no sólo en el interior de la propia FIR.

Un proyecto tan ambicioso de comunicaciones interordenador hasta ahora sólo se ha adoptado en Estados Unidos, donde una sola administración, la Federal Aviation Administration, coordina el control aéreo en los 51 estados.

En lo referente a la configuración hardware y software de los sistemas descritos, es claro que las prestaciones requeridas por las máquinas dependen fuertemente del nivel de automatización que se quiera alcanzar.

Típicamente, las automatizaciones radar prevén el empleo de calculadores medios (incluso son suficientes los de 16 bits), sin necesidad de me-

moria masiva. Los datos radar se procesan previamente con sofisticados aparatos llamados extractores, con lo que la misión del procesador se limita al desarrollo de una serie de cálculos sobre un restringido número de datos.

Aunque el tratamiento de los planes de vuelo y de los planes repetitivos (RPL) no prevé el empleo de grandes ordenadores, en cambio son necesarias grandes bases de datos para la geografía, para las características de los aviones y para la memorización de los propios RPL. Si el sistema de análisis de los planes de vuelo y de la impresión de las strips se hace fuera de línea (no integrado con los datos radar) es posible disponer todos los datos necesarios en el disco, porque los tiempos de I/O sobre estos aparatos no son críticos en estas condiciones. En cambio, los niveles de control integrado prevén sistemas más complejos para elevar la velocidad de cálculo y la capacidad de la memoria real o de masa. Estos representan una aplicación típica del llamado tiempo real, puesto que deben adquirir periódicamente los trazados y procesarlos para la presentación en sincronismo con la rotación del radar (menos de 10 segundos por revolución).

Para la realización de los complejos cálculos o las largas exploraciones de las bases de datos entre una entrada Y otra, los calculadores deben permitir la ejecución de por lo menos un millón de instrucciones por segundo (1 mips). Para asegurar la continuidad de empleo es necesaria la redundancia de los componentes del sistema, para eliminar las consecuencias de los fallos de hardware o de software.

Las causas de «caída» de un sistema pueden ser muchísimas, aunque se reúnen en tres categorías: fallos de alimentación, fallos de hardware y fallos de software.

En el primer tipo entran las potenciales faltas de suministro de energía eléctrica. La solución es la clásica de los grupos electrógenos (también redundantes) soportados por baterías tampón para suplir la ausencia de energía entre el fallo de alimentación y la puesta en marcha de los grupos, puesto que bastan pocas décimas de segundo para perder las informaciones contenidas en las memorias.

El segundo tipo de fallo corresponde a los componentes del sistema electrónico y electromecánico que solemos llamar ordenador.

El restablecimiento de un fallo de hardware es un poco más lento (aproximadamente 1 minuto)

que el que sigue a un fallo de alimentación, aunque también es completamente automático. Para no perder todas las informaciones introducidas por los diversos controladores, en el momento de la introducción de cada comando, en disco se hace una copia de las bases de datos contenidas en la memoria real. Cuando un ordenador sufre un fallo, el ordenador de reserva lo advierte y puede acceder a los mismos discos. En el momento de su puesta en marcha carga en sus memorias todas las informaciones correspondientes al tráfico en curso.

En Italia, el proyecto de los sistemas automatizados de control del tráfico aéreo aparece a fines de los años 60 y al principio de los años 70, cuando, a petición de la Aeronáutica Militar Italiana, se proyectó el sistema ATCAS (Air Traffic Control Automatic System), fruto de la colaboración entre Selenia e IBM, para el control de la región de vuelo de Roma. El programa preveía el suministro en 10 años (del 75 al 85) de todos los niveles de automatización descritos.

Desde 1980, Datamat Ingegneria dei Sistemi de Roma se ha unido al consorcio para el proyecto y el desarrollo del software de los últimos niveles de automatización.

En la configuración final, el sistema prevé el empleo de un main-frame del tipo IBM 3033, que tiene una velocidad de ejecución de 4.5 mips. Este está conectado a dos GP-16, a un CDG (Selenia) y a un segundo IBM 3033 que tiene la función de «esclavo» (gráfico de al lado).

Los principales periféricos gestionados por el main-frame son los siguientes:

- unidades de disco de 400 Mbytes cada una, para la memorización de la situación actual y para el archivo de los planes repetitivos
- unidades de cinta para el registro legal de todas las comunicaciones entre el controlador y el sistema
- unidades impresoras para la impresión de las strips y para la copia en papel de los telex recibidos por la red AFTN
- unidades de vídeo para la corrección de los mensajes télex y para la inserción de planes de vuelo o de planes repetitivos

La gestión de los monitores (PP/EDD) a disposición de los controladores la realizan los GP-16, que reciben del main-frame los datos procesados. Estos minis también actúan como interfaces entre los operadores y el amo, gestionando

```

(FPL-CLUESO-IN-C12/L-S/C
-LIRN1530 LQV1845
-245F180 TEA1 TEA A14 FRZ A12 MUN MLD SDS
-EDOC1990
-OPR US MIL REG22550 AUSTRIA DIP CLNCS6676 REG ATD TO EDOC2P
-FUEL0500 P084)

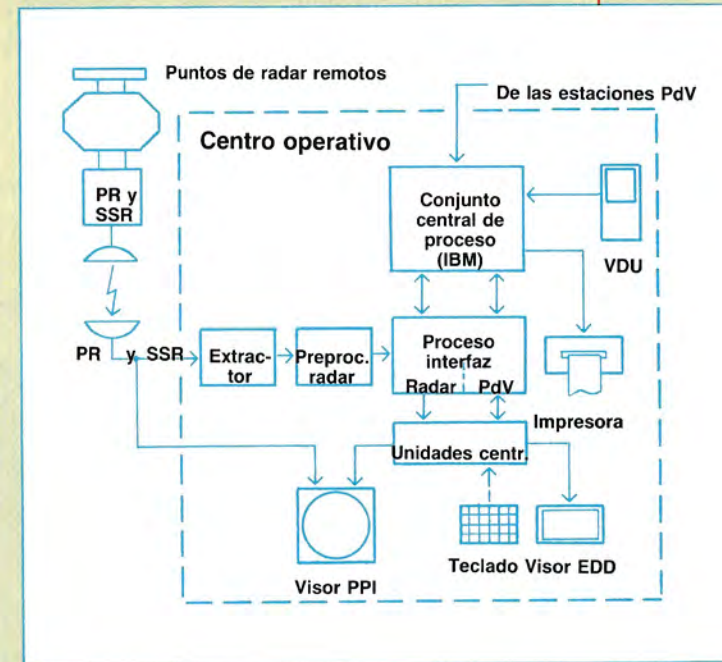
10317 261502

CDR **C15*E02*A/D NO CONGIUNG**INF **C09* ** * TIPO A/TI SCOR **
INF **C15*E01* FLTAS ASSUNTA **

-NESSUNA ROTTA

(FPL-CLUESO-IN-C12/L-S/C
-LIRN1530 LQV1845
-0350F250 <(TEA1)> TEA A14 FRZ A12 MUN MLD SDS
-EDOC1990 -0)
```

B. Liotta



Arriba, presentación en monitor de los mensajes transmitidos por la red AFTN. Debajo, el sistema de adquisición

los teclados normales o los dispositivos rápidos de entrada.

En el caso de un fallo de hardware o de software del IBM 3033, el CDG está preparado para proporcionar un labelling de los datos radar a los controladores durante el minuto necesario para el restablecimiento automático.

Umberto Corà y Bruno Liotta,
Datamat Ingegneria dei Sistemi, Roma

Así se trazan los diversos histogramas, siempre como columnas llenas, variando únicamente el color. En el vídeo monocromático, las columnas deben dibujarse de manera diferente.

El programa está dividido en dos partes principales, la primera dedicada a la introducción de los datos y la segunda a la representación. La introducción está incluida en el main para poder variarse o excluirse fácilmente si se desea adaptar el programa a empleos generalizados; en cambio, la otra sección está constituida por una serie de rutinas, cada una dedicada a una tarea específica. En el listado no se ha hecho la división en subrutinas, puesto que las diversas funciones son realizadas una a continuación de otra; sin embargo, en los diagramas de flujo de las páginas 1519 a 1522 se ha representado la numeración habitual. Para dar al programa una forma más estructurada es necesario interrumpir el flujo principal antes de la rutina 1000 insertando las llamadas, y terminar cada bloque a aislar (subrutina) con la instrucción RETURN.

De ahora en adelante, los diferentes bloques serán considerados como subrutinas.

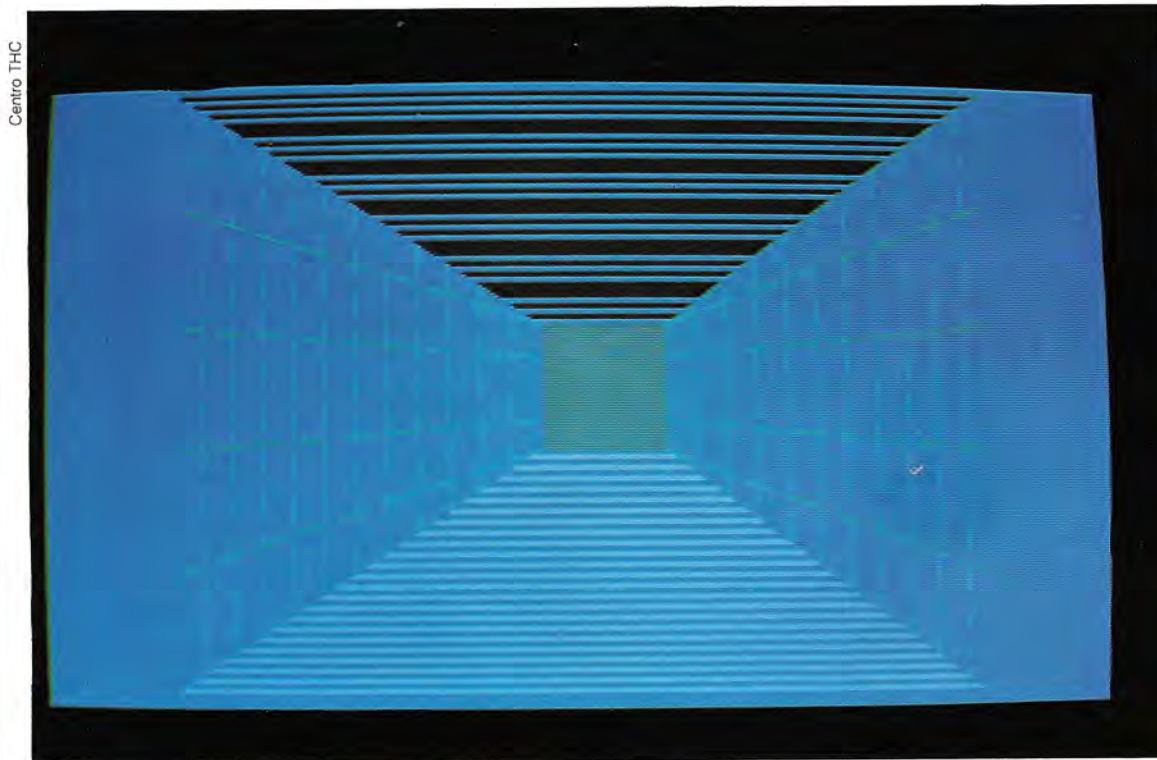
El programa también contiene las subrutinas de presentación de los caracteres alfabéticos y nu-

méricos; las correspondientes DATA son inicializadas en las líneas 60 y 65. Sigue la fase de entrada de datos, durante la cual se realiza el control de validación: en este caso se ha elegido la introducción de valores en porcentaje, que no pueden superar el valor 100 (100%) y también la suma de los valores introducidos (línea 210; la subrutina 6000 sólo contiene la emisión de la escritura del diagnóstico). La salida del programa puede obtenerse introduciendo un número de datos igual al máximo previsto (NM = 20, línea 20) o introduciendo un valor negativo cualquiera. El límite máximo de 20 datos, correspondientes a 20 columnas a distribuir en el eje X, se debe a las dimensiones de la pantalla utilizada y a las particulares formas de llenado de los histogramas. Para el transporte del programa a otras máquinas deben sustituirse las instrucciones específicas del ordenador utilizado en el ejemplo. En particular:

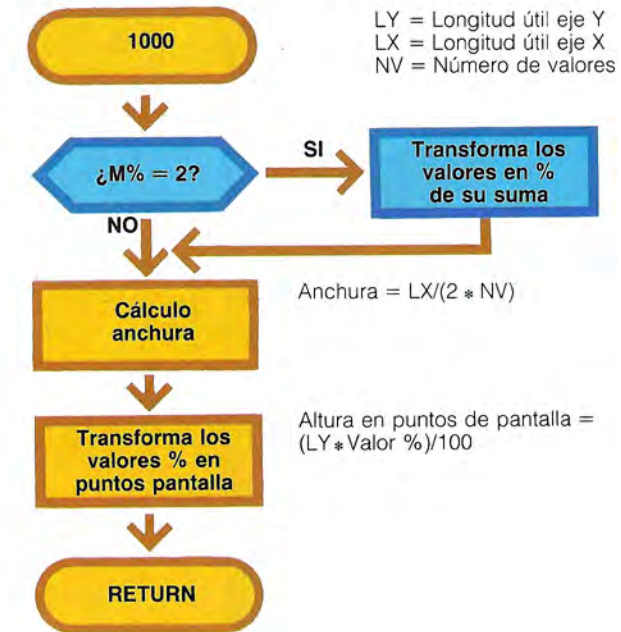
TEXT

indica el modo de texto; se utiliza al principio del programa para desactivar el modo gráfico

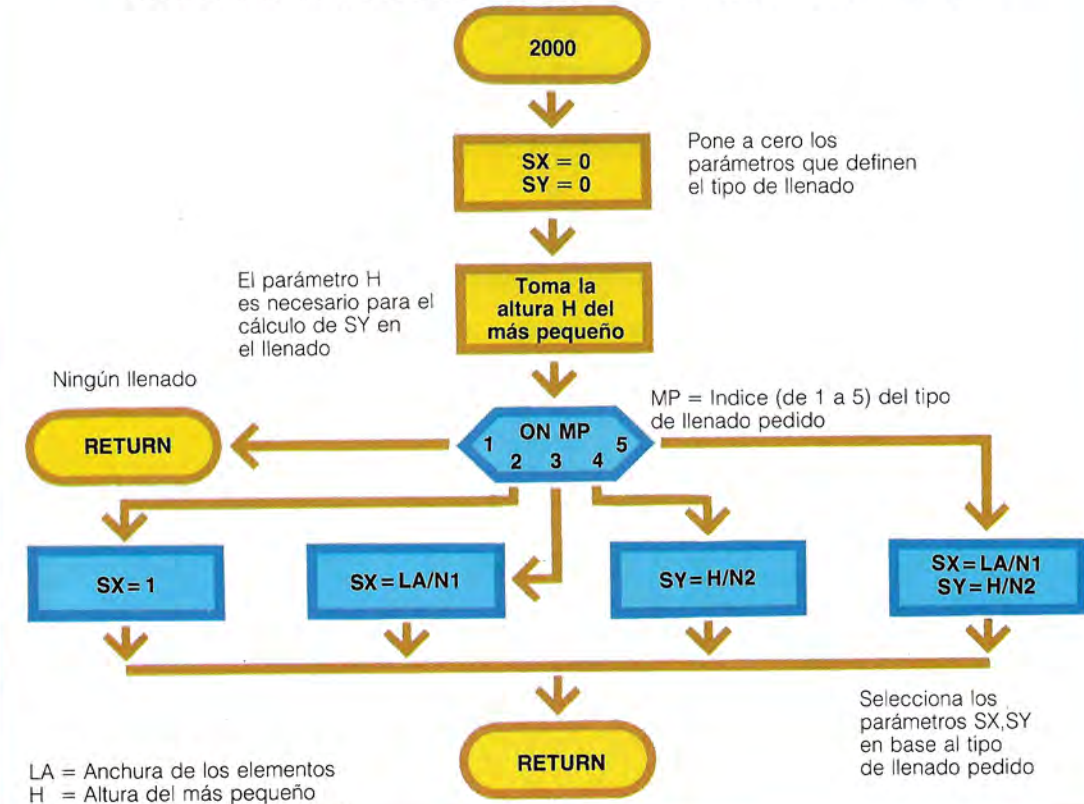
Simulación en perspectiva de un ambiente tridimensional en un monitor de colores.



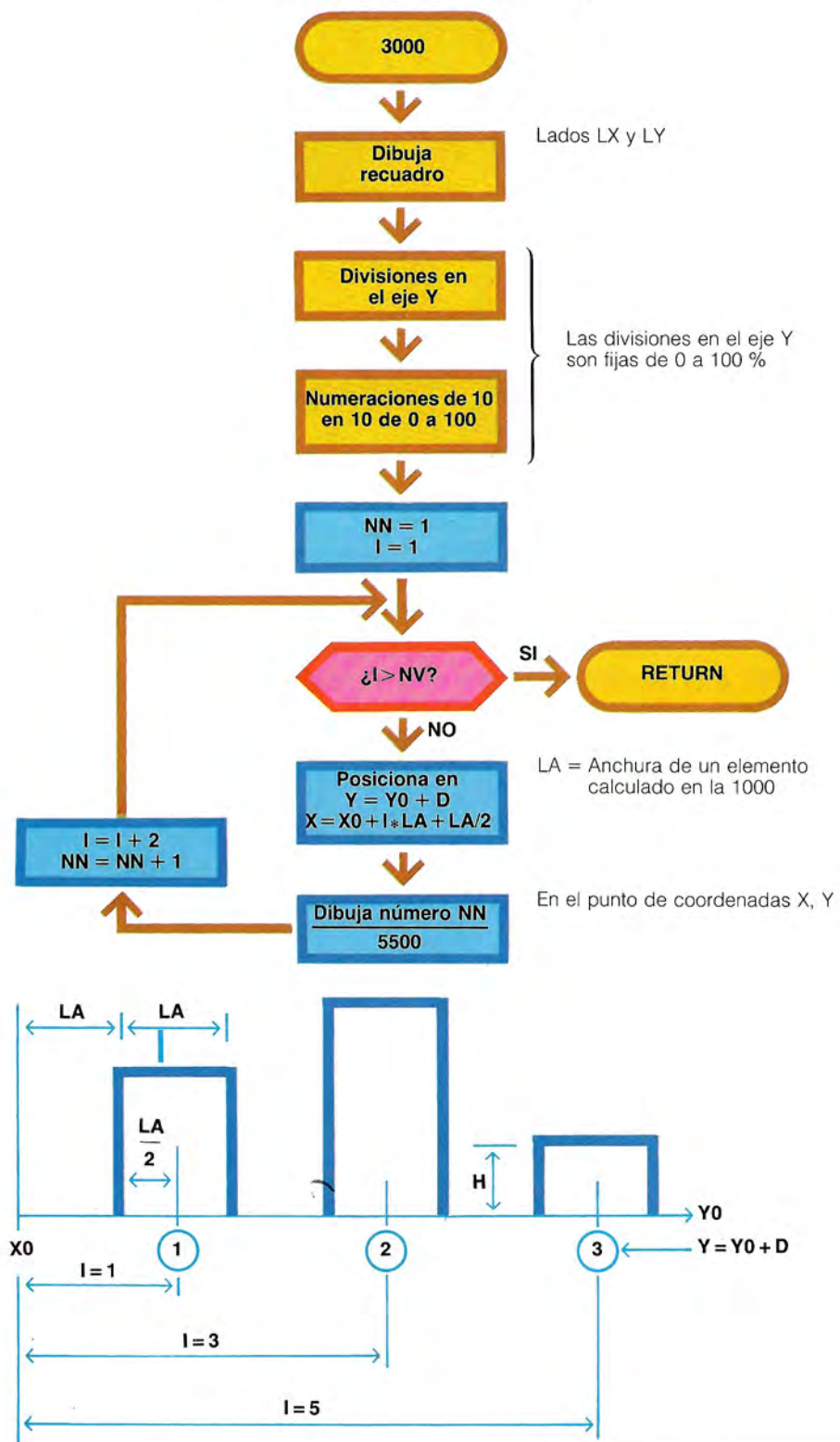
PROCESO DE LOS DATOS PARA EL HISTOGRAMA



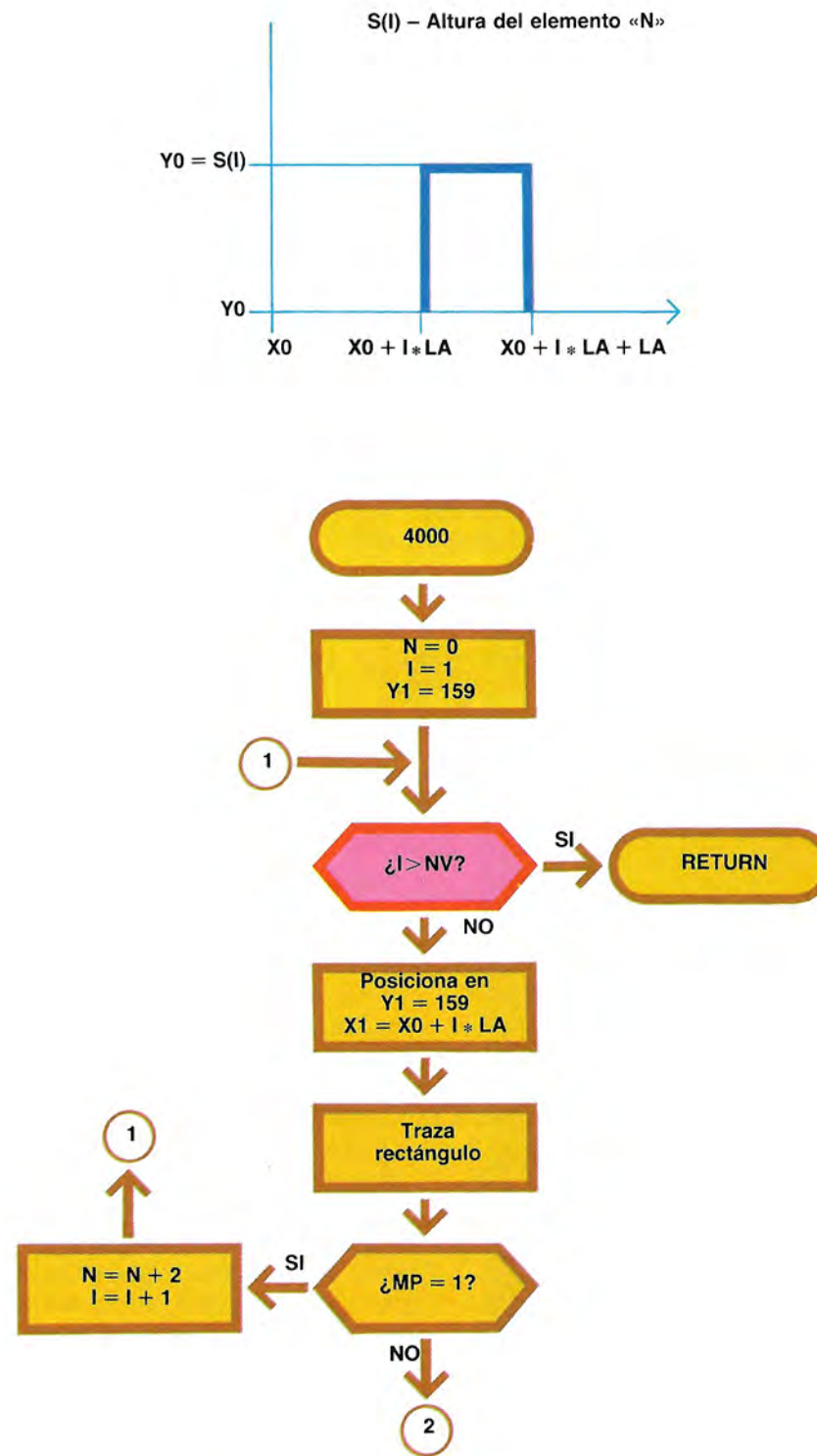
IMPLANTACION DE LOS PARAMETROS DE PRESENTACION

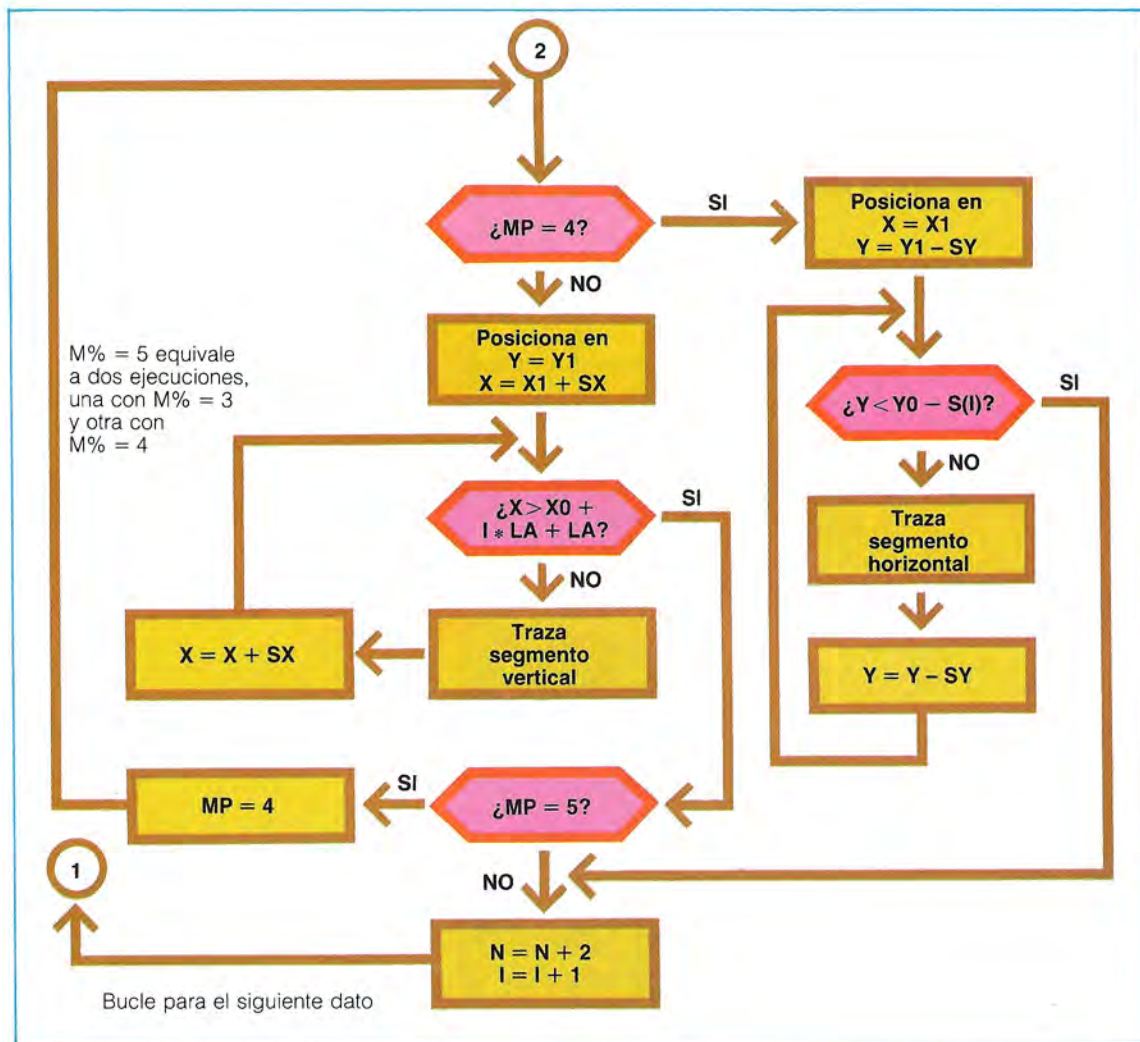


REPRESENTACION VENTANA



REPRESENTACION HISTOGRAMA





HOME
VTAB n
HTAB m
HGR2
HCOLOR = n
HPLOT X1,Y1 TO X2,Y2
GET A\$

borra el contenido de la pantalla y posiciona el cursor arriba a la izquierda
posiciona el cursor en la línea n
posiciona el cursor en la columna m
activa el modo gráfico en alta resolución activa la «pluma» de color n
une con un segmento los puntos X1,Y1 y X2,Y2
adquiere un carácter del teclado; es la equivalente a la instrucción INPUT\$(1).

En el programa, todos los posicionados se refieren al origen, que está arriba a la izquierda. Para diferentes posicionados puede ser necesario modificar el signo de los desplazamientos. El diagrama de flujo de un programa que puede presentar diagramas de tarta se ha representando en las páginas 1525 y 1526. Este programa no presenta diferencias conceptuales con respecto al de los histogramas. En lugar de representar una serie de rectángulos de altura proporcional a los valores de los datos a representar (o a los porcentajes sobre el total que representan estos valores), el programa debe presentar un círculo dividido en tantos sectores como cuantos son los datos a representar, atribuyendo a cada sector un área proporcional al dato que representa. En el programa, cuyo listado puede verse en las páginas 1527 y 1528, se ha previsto el empleo de un monitor en colores.

REPRESENTACION DE HISTOGRAMAS

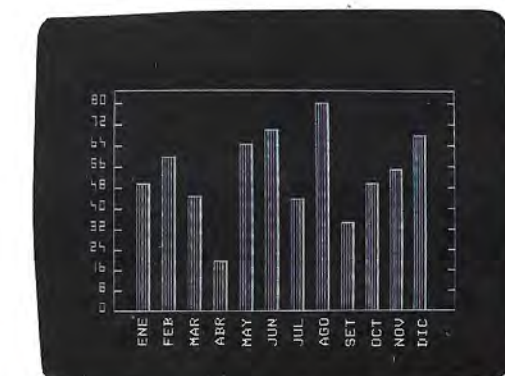
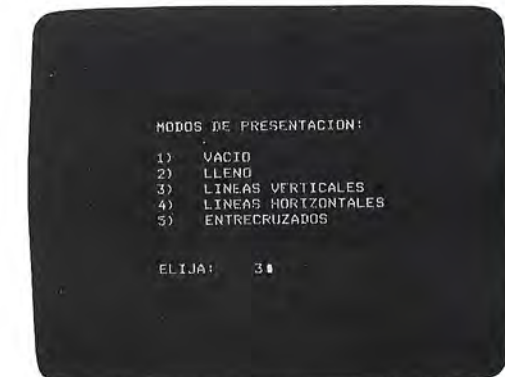
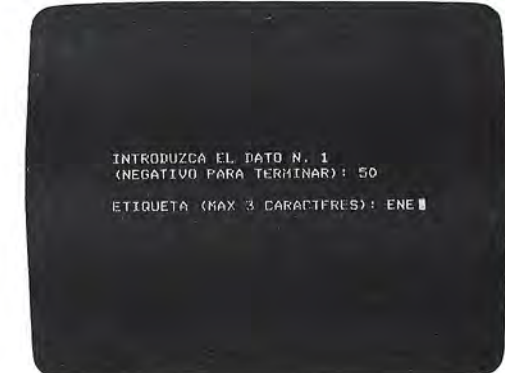
Representamos aquí algunas fases del funcionamiento del programa de representación de los histogramas.

Después del coloquio gestionado por la línea 80, en que el usuario ha declarado que introducía datos absolutos (no en porcentajes), el programa ha pasado a la fase de introducción de los datos a representar. El usuario ha introducido aquí el valor 50 como primer dato, que se contraseña con la etiqueta 50.

Las líneas 310 a 360 presentan, al final de la fase de introducción, el menú de las formas de representación posibles. El usuario ha elegido la representación con columnas de líneas verticales.

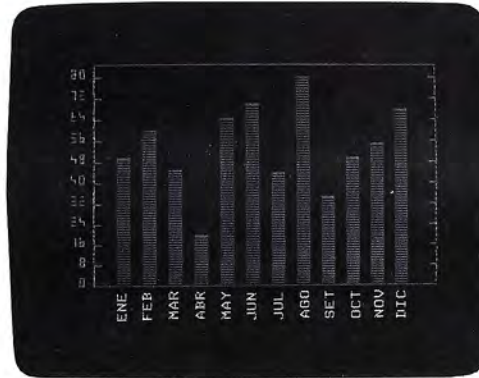
El programa pide después si se desea representar los datos en forma de porcentaje. La respuesta del usuario es negativa.

El programa ha activado la representación en líneas verticales de los valores absolutos introducidos.

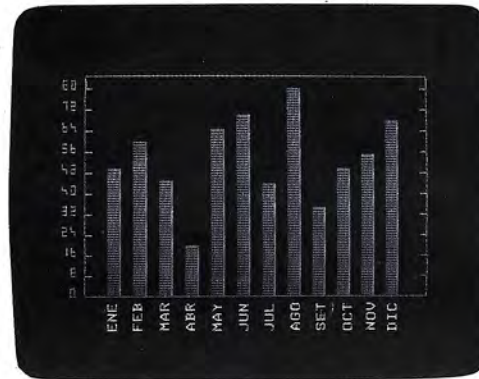


En la foto de al lado y en las tres fotos que siguen se han representado los posibles modos de presentación.

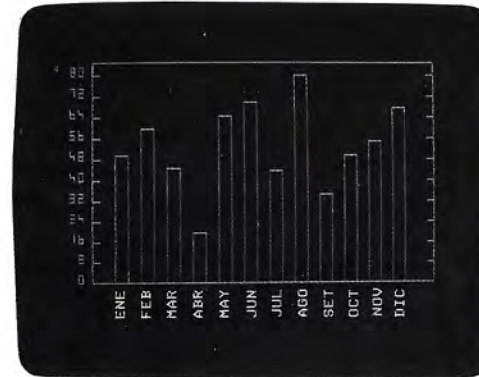
En líneas horizontales...



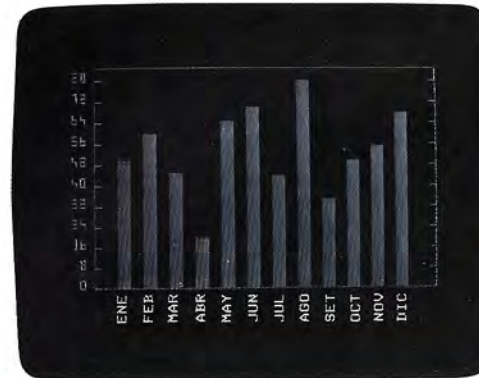
En líneas entrecruzadas...



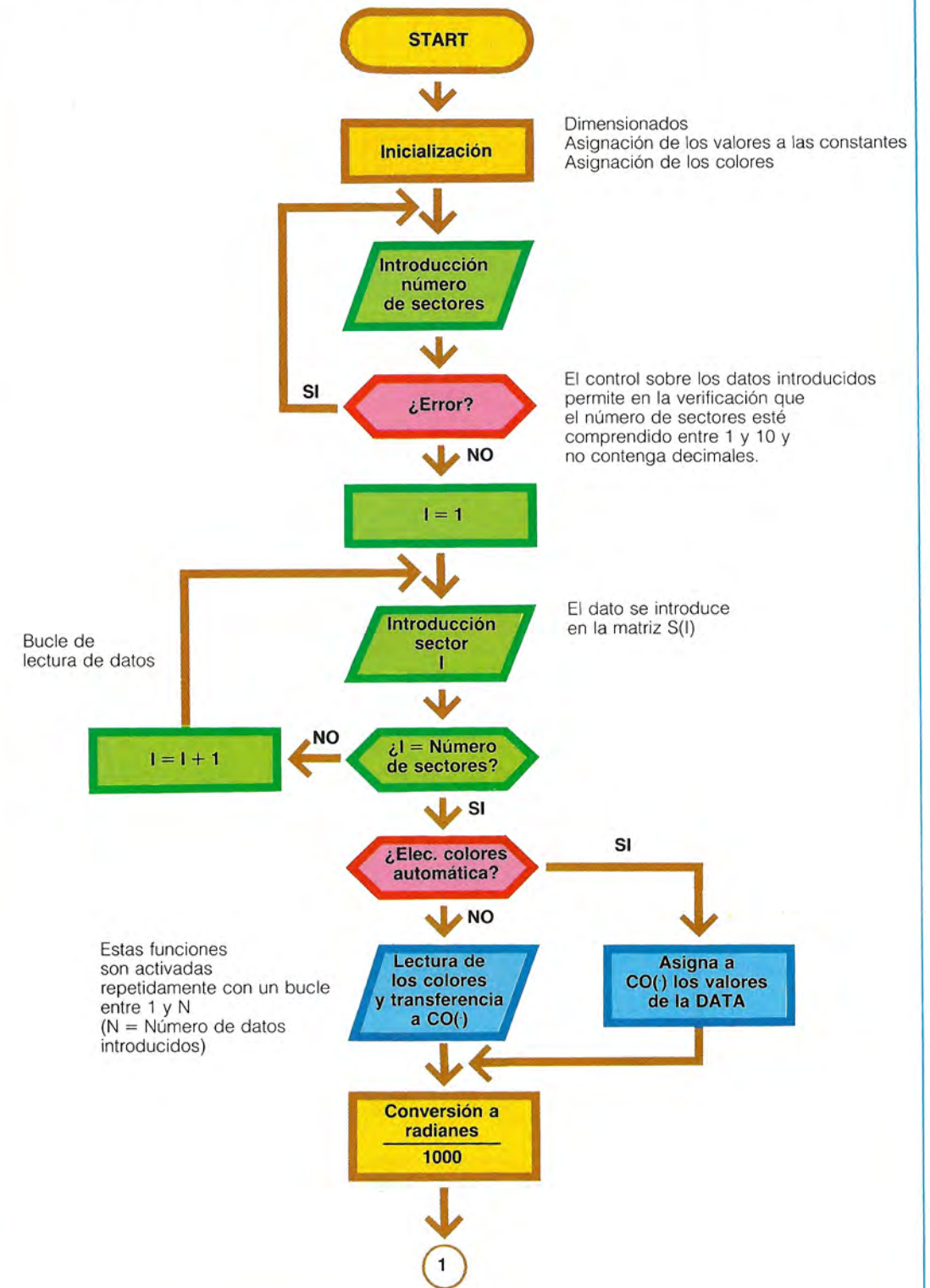
En columnas vacías...



En columnas llenas...



PROGRAMA PARA LA PRESENTACION DE DIAGRAMAS DE TARTA

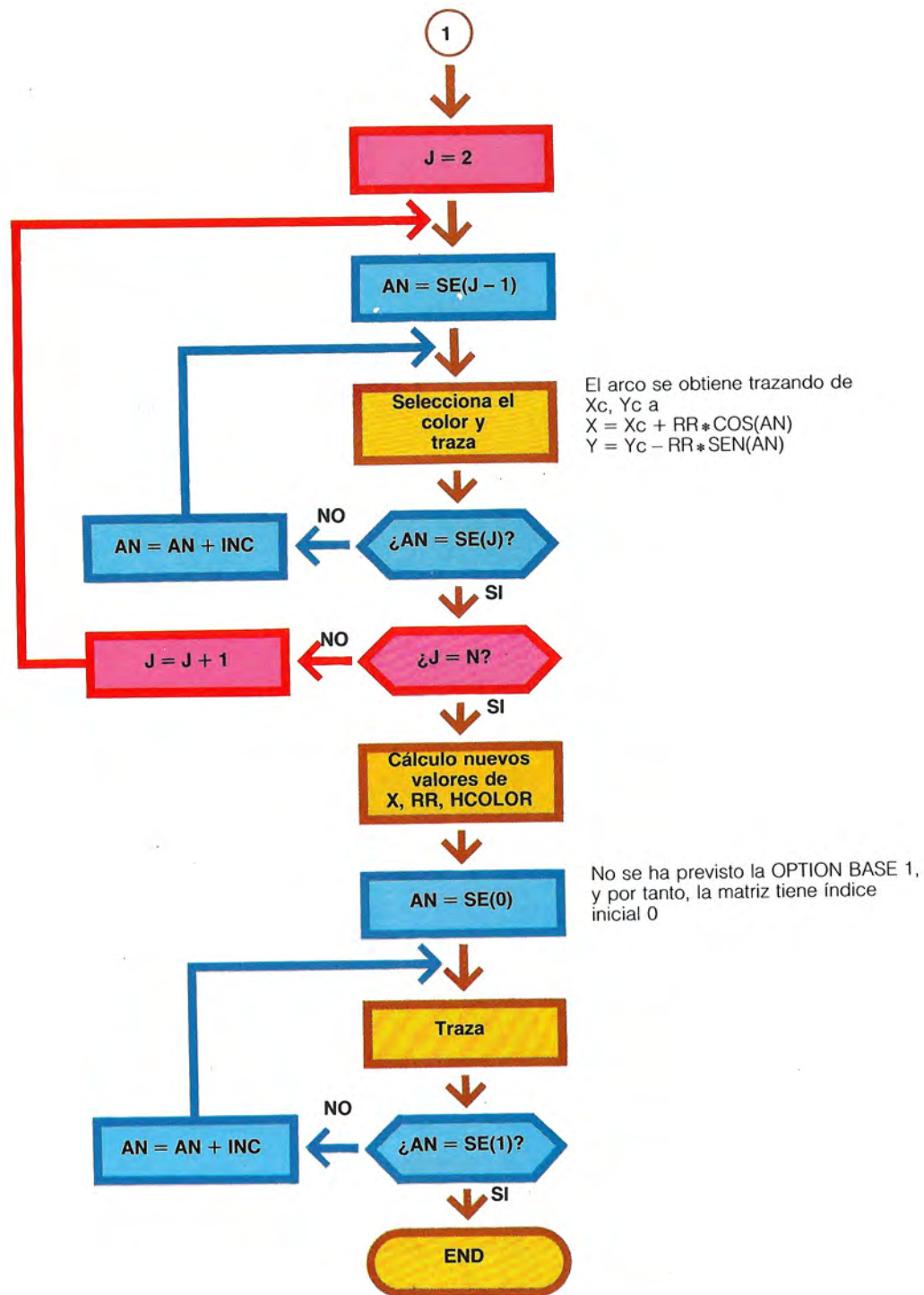


PRESENTACION DE DIAGRAMAS DE TARTA

```

10  REM -----
20  REM  DIAGRAMAS DE TARTA
30  REM -----
40  REM
50  REM
60  XC = 130
   YC = 90
   R = 90
   PI = 3.1415926
   INC = PI / 300
70  DIM SE(10),C(10),CO(10),S(10)
80  FOR I = 1 TO 10
   : READ C(I)
   : NEXT
90  DATA 5,6,7,1,2,3,5,6,7,1
100 REM
110 REM -----
120 REM  ENTRADA DATOS
130 REM -----
140 REM
150 HOME
160 VTAB 4
   : INPUT "CUANTOS SECTORES? (MAX 10)"
   : A$
170 A = VAL (A$)
180 IF A < 1 OR A > 10 OR INT (A)
   < > (A) THEN 150
190 VTAB 6
   : PRINT "INTRODUCIR LOS DATOS (NUMEROS
   POSITIVOS PARA CADA SECTOR."
200 N = A
   : TT = 0
210 FOR I = 1 TO N
220 VTAB (10 + I)
   : PRINT "SECTOR "I;
   : INPUT " ";A$
230 S(I) = VAL (A$)
   : IF S(I) <= 0 THEN 220
240 TT = TT + S(I)
   : S(I) = TT
250 NEXT
260 HOME
270 VTAB 4
   : PRINT "LA TARTA TIENE "N" SECTORES."
280 REM
290 REM -----
300 REM  SELECCION COLORES
310 REM -----
320 REM
330 VTAB 6
   : INPUT "QUIERE SELECCIONAR LOS COLORES
   O QUIERE QUE LO HAGA EL ORDENA
   DOR? (1/2) ";A$
340 IF A$ < > "1" AND A$ < > "2"
   THEN 260
350 IF A$ = "1" THEN 390
360 FOR I = 1 TO N
   : CO(I) = C(I)
   : NEXT
370 IF CO(N) = 5 THEN CO(N) = 1
380 GOTO 550
390 HOME
400 VTAB 4
   : PRINT "ELIJA ENTRE VERDE, ROJO,
   ANARANJADO, AZUL Y BLANCO TE
   CLEANDO <V>,<R>,<A>,<B> O <W>"
410 FOR I = 1 TO N
420 VTAB (6 + I)
   : PRINT "COLOR DEL SECTOR "I;
   : INPUT " ";A$
430 IF A$ = "V" THEN CO(I) = 1
   GOTO 490

```




```

440 IF A$ = "R" THEN CO(I) = 2
: GOTO 490
450 IF A$ = "A" THEN CO(I) = 3
: GOTO 490
460 IF A$ = "R" THEN CO(I) = 4
: GOTO 490
470 IF A$ = "W" THEN CO(I) = 5
: GOTO 490
480 GOTO 420
590 NEXT
500 REM
510 REM -----
520 REM PRESENTACION SECTOR
530 REM -----
540 REM
550 HOME
560 VTAB 4
: INPUT "QUIERE PRESENTAR EL SEC
TOR 1? (S/N) ";A$
570 FL = 0
: IF LEFT$(A$,1) = "S" THEN FL
= 1
580 GOSUB 1000
590 REM
600 REM -----
610 REM DIBUJO DE LA TARTA
620 REM -----
630 REM
640 HGR2
650 FOR J = 2 TO N
660 FOR AN = SE(J - 1) TO SE(J)
STEP INC
670 HCOLOR= CO(J)
680 HPLOT XC, YC TO XC + R * COS (AN),
YC - R * SEN (AN)
690 NEXT AN, J
700 X = XC + 15 * FL
: HCOLOR= CO(1)
: RR = R - 8 * FL
710 IF SE(1) > .5 THEN RR = R - 3
* FL
: X = XC + 12 * FL
: IF SE(1) > 1 THEN RR = R
: XC = XC + 9 * FL
720 FOR AN = SE(0) TO SE(1) STEP INC
730 HPLOT X, YC TO X + RR * COS (AN), Y
C = RR * SEN (AN)
740 NEXT
750 END
1000 REM
1010 REM -----
1020 REM CONVERSION EN RADIANES
1030 REM -----
1040 REM
1050 IF TT = 360 THEN 1090
1060 FOR I = 1 TO N
1070 S(I) = S(I) / TT * 360
1080 NEXT
1090 FOR I = 1 TO N - 1
1100 SE(I) = S(I) - S(1) / 2
1110 NEXT
1120 SE(0) = - S(1) / 2
SE(N) = 360 + SE(0)
1130 FOR I = 0 TO N
1140 SE(I) = SE(I) / 180 * PI
1150 NEXT
1160 RETURN

```

Los colores de los sectores son elegidos por el programa, a menos que el usuario lo haga. La única particularidad a observar es la rutina 1000, que convierte los datos introducidos por

el usuario en valores angulares (anchuras de los sectores) expresados en radianes. Para el resto, las instrucciones son análogas a las ya utilizadas para los histogramas.

DIAGRAMAS DE TARTA

El programa (línea 160) pide al usuario cuántos sectores (cuántos datos) debe representar. A la introducción del número de sectores (en este caso 5) le sigue la fase de petición de datos (líneas 190 a 250).

```

CUANTOS SECTORES? (MAX 10) 5
INTRODUZCA LOS DATOS (NUMEROS POSITIVOS)
PARA CADA SECTOR

SECTOR 1: 10
SECTOR 2: 5
SECTOR 3: 50
SECTOR 4: 25
SECTOR 5: 15

```

Al final de la introducción, el programa informa al usuario acerca del número de datos introducidos (línea 270) y pide aclaraciones sobre la elección de los colores (línea 330).

```

LA TARTA TIENE 5 SECTORES
QUIERE ELEGIR LOS COLORES O QUIERE QUE LO
HAGA EL ORDENADOR? (1/2) 2

```

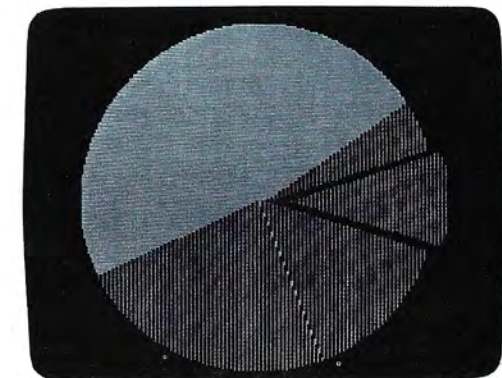
Como el usuario ha activado la selección automática de los colores, el control pasa a la línea 560, que pide si se quiere presentar el primer sector. La respuesta es afirmativa.

```

QUIERE PRESENTAR EL SECTOR 1? (S/N) S

```

Presentación del diagrama de tarta. El sector «despegado» de la tarta es el primero: los otros le suceden en sentido antihorario.



Empleo de la memoria en el funcionamiento en modalidad gráfica

El monitor es la unidad de salida más utilizada en las aplicaciones de gráficos de ordenador. Por su naturaleza, este dispositivo no conserva memoria de las imágenes que aparecen en él; la presentación de una leyenda o de un dibujo, que en el programa se obtiene con una sola instrucción (PRINT, LINE, etc.), en realidad necesita una continua actividad de revisualización, a intervalos muy cortos, de la misma imagen para reactivar cada punto iluminado antes de que desaparezca su luminosidad. Por este motivo, la imagen gráfica debe memorizarse en una zona de memoria dedicada; a intervalos regulares, las rutinas del sistema toman el contenido de esta zona y lo transfieren a la pantalla, redibujando continuamente el gráfico. Esto también es válido para cualquier leyenda en Basic o en otros lenguajes. De este funcionamiento surge la necesidad de disponer de una zona de memoria que conserve el estado de cada punto de la pantalla. En una pantalla monocromática basta un solo bit para cada punto: el bit cero indica, por ejemplo, que el punto no es visible, el bit no indica que el punto de pantalla está activado. En los monitores en color, la extensión de la memoria dedicada es mayor, porque para cada punto, además del estado (ON/OFF), debe definirse también el color. Por tanto, la presentación de un dibujo se produce cargando la imagen binaria en la memoria gráfica y activando el proceso de presentación. Por ejemplo, las instrucciones que sirven para trazar un segmento entre dos puntos (LINE, PLOT, etc.) calculan de las coordenadas en puntos de pantalla las correspondientes posiciones de memoria y las activan, poniendo en estado ON todos los bits correspondientes al segmento a trazar. En una máquina de 8 bits, cada posición de memoria puede controlar como máximo 8 puntos de pantalla. Escribiendo en una de ellas el valor 1 se activa el primero de los puntos controlados, con el valor 3 se activan dos puntos contiguos (1 + 2), y así sucesivamente hasta el valor máximo previsto, que activa el máximo número de puntos de pantalla contiguos. La calidad gráfica de un dibujo depende estrechamente de la resolución de la pantalla, pero el aumento de los puntos aumenta la zona de memoria necesaria; por tanto, cuando la pantalla del vídeo permite

una elevada resolución, debe reducirse el número de puntos gestionados para no ocupar una excesiva cantidad de memoria. Por ejemplo, una pantalla dividida en 280 posiciones horizontales y 190 verticales necesita unos 53.000 indicadores de estado, cada uno de los cuales está constituido por 1 bit; por tanto, se tiene una ocupación de memoria de 7,5 kbytes en el supuesto de utilizar solamente 7 de los posibles 8 bits de cada posición.

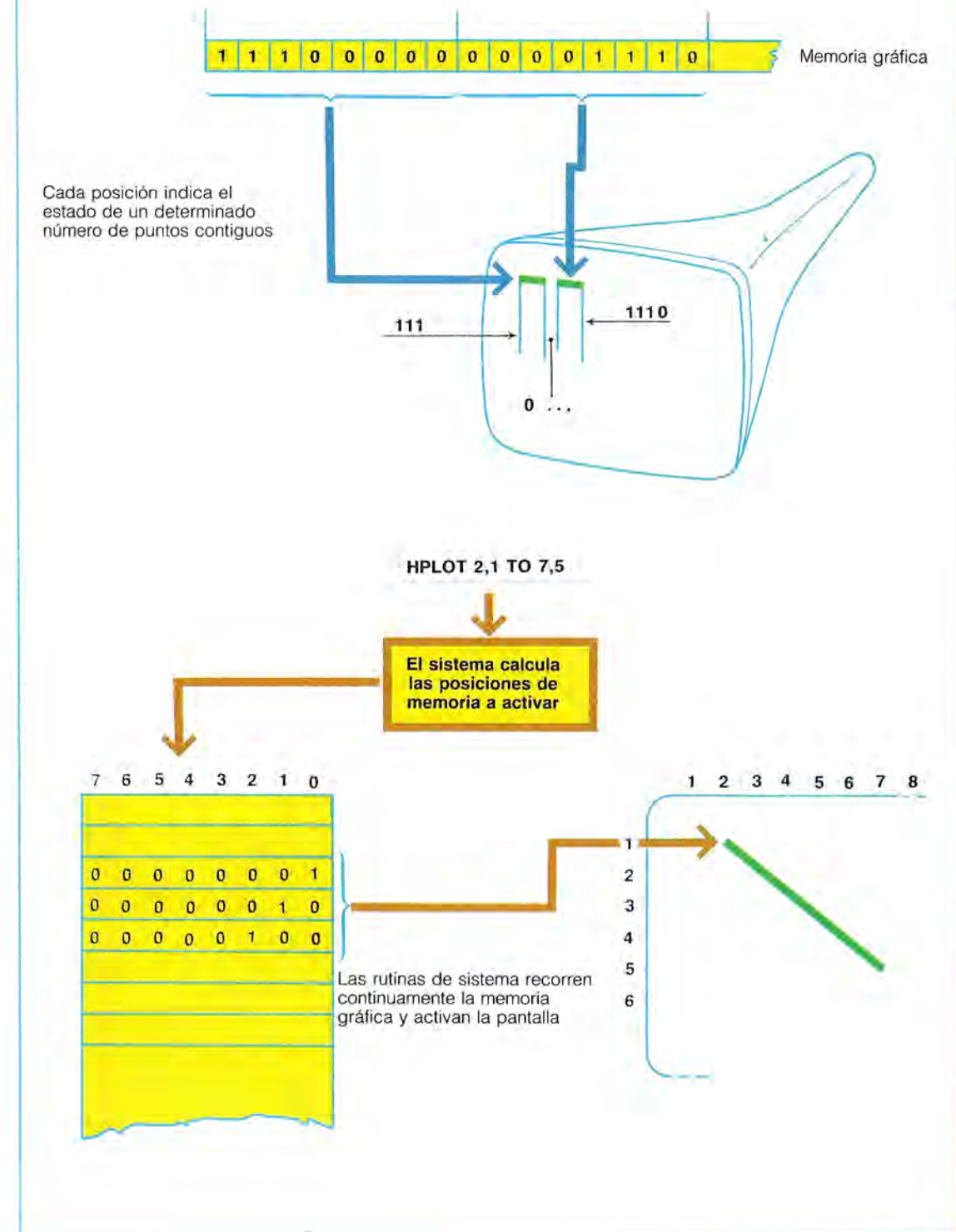
La existencia de esta zona de memoria dedicada ofrece en compensación notables posibilidades de programación. Para dibujar figuras de una cierta complejidad puede aprovecharse el acceso directo a las posiciones de memoria sin estar vinculadas a las instrucciones gráficas, o viceversa, puede leerse el contenido de la memoria para conocer el estado de cada punto del vídeo (1 para iluminado, 0 para apagado) y para procesar o memorizar el dibujo. Es decir, este método prevé considerar el gráfico como una matriz de valores numéricos sobre la cual puede realizarse cada tipo de presentación. La presentación en el vídeo puede producirse automáticamente, a medida que la matriz se modifica, o con comandos cuando se activa el modo gráfico. Este método de gestión no es común a todas las máquinas. En algunas, la memoria gráfica es accesible sólo para el sistema, y en este caso, deben utilizarse siempre instrucciones de alto nivel.

La memoria vídeo

La relación que liga un punto de la pantalla vídeo a la posición de memoria que la gestiona, casi nunca es sencilla ni lineal. El mecanismo de gestión de la pantalla vídeo está influenciado por diversos parámetros, también dependientes del hardware de la máquina. A continuación ilustraremos un ejemplo de distribución de memoria empleada frecuentemente en los ordenadores personales y microordenadores que se sirven de la CPU Rockwell 6502, o sea del tipo Apple y compatibles. Los valores proporcionados en los ejemplos se refieren a máquinas particulares; para las otras, puede ser necesario modificar los valores numéricos.

La zona de memoria dedicada a la pantalla casi nunca ocupa posiciones contiguas, principalmente por problemas de hardware. De esta manera, dos zonas del vídeo que distan, por ejemplo 100 posiciones, no corresponden a dos memorias cuyas direcciones difieren en 100.

GESTION SIMPLIFICADA DEL VIDEO EN MODO GRAFICO



En la figura de la página siguiente se ha representado una de las dos páginas de memoria vídeo utilizadas normalmente en los sistemas que adoptan el 6502.

La primera zona de la pantalla (7 puntos) tiene como memoria imagen la posición 8192; las zonas que siguen sobre la misma línea tienen memorias contiguas, por lo que para seleccionar las varias zonas sobre la misma línea basta con añadir 1 a la dirección inicial. La pantalla está dotada de 40 columnas (normalmente numeradas de 0 a 39) y, por tanto, la primera línea (numerada 0) está controlada por las posiciones de memoria comprendida entre la dirección 8192 y la $8192 + 39 = 8231$. Pasando a la segunda línea (la número 1) podría esperarse que la memoria correspondiente empieza por la primera posición libre a continuación de la última ocupada por la línea anterior, o sea la posición 8232. En realidad, esta memoria corresponde a un punto de la pantalla mucho más bajo. La segunda línea (la número 1) en cambio empieza en la posición 9216, que dista 1024 de la memoria del principio de la línea anterior. También para esta línea, como para todas, cada zona de la pantalla corresponde a una memoria contigua. El incremento 1024 permanece válido para siete

líneas (de la número 1 a la 7), mientras que la dirección de la octava línea es igual a la de la primera (la número 0) más 128. Este mecanismo se repite siete veces para formar ocho grupos de ocho líneas cada uno. Nuevamente, la segunda mitad del vídeo tiene un mapa dividido en 8 grupos de 8 líneas; sólo varía la dirección de la primera posición, que en este caso específico es 8232.

La figura vuelve a asumir las particiones de memoria de todo el vídeo. La primera parte se muestra por entero y la primera sólo se indica, puesto que los mecanismos de apuntado de las direcciones se regulan con el mismo principio. El primer punto del vídeo, identificado por las coordenadas 0,0, corresponde a la posición de memoria 8192, más precisamente al bit 0 de esta posición. El valor 8192 es la referencia con respecto a la cual se calcularán las direcciones de todos los demás puntos.

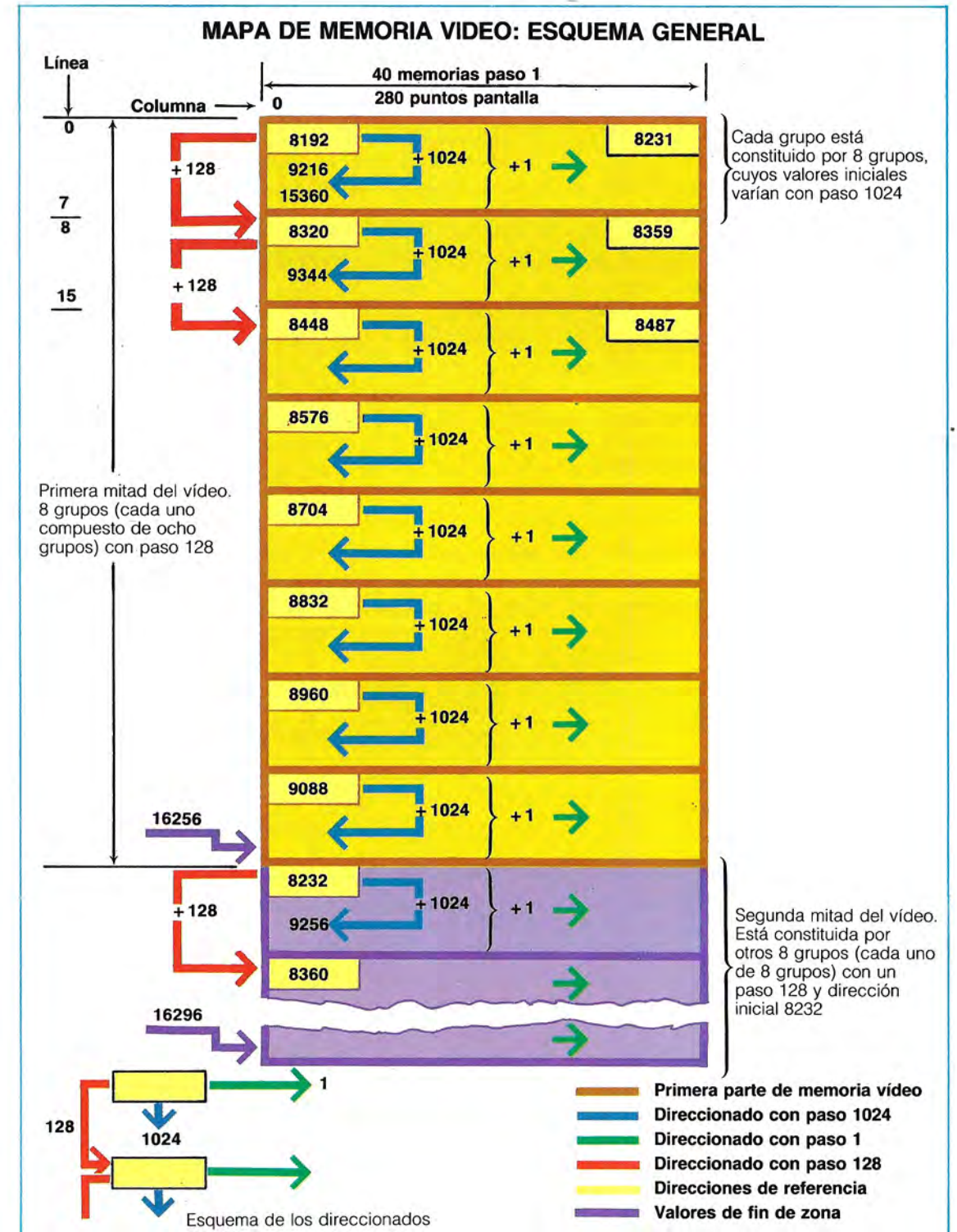
Direccionado de la memoria vídeo

Dada la complejidad del mecanismo de direccionado, consideraremos ahora un ejemplo de cálculo y un programa utilizable como subrutina de conversión entre posiciones de memoria y puntos de pantalla, o viceversa.

El ordenador para gráficos también encuentra ocasiones de empleo en el sector artístico.



Centro THC

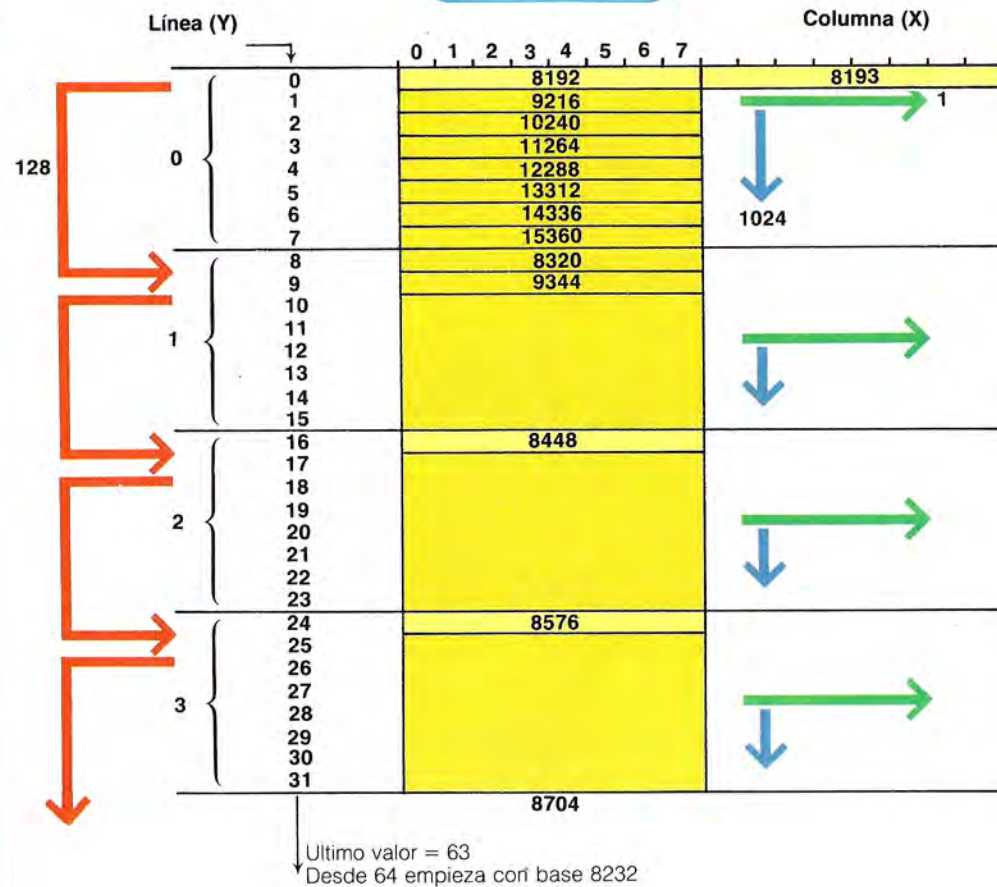
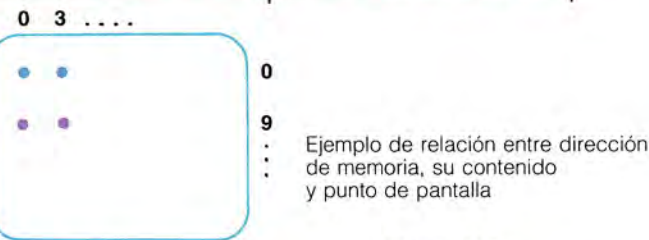


Direccionado según el eje Y. Supongamos que se quiere direccionar el punto en la columna cero ($X = 0$) perteneciente a una línea Y cualquiera. La variable de entrada es la ordena-

da (Y), y de este valor debe obtenerse la correspondiente posición de memoria. Por ejemplo, con $Y = 7$, la correspondiente posición es 15360 (figura de arriba).

DIRECCIONADO EJE Y

Coordenadas		Posiciones de memoria	Contenido de la memoria							
X	Y		0	1	2	3	4	5	6	7
3	0	8192	0	0	0	1	0	0	0	0
5	0	8192	0	0	0	0	0	1	0	0
3	9	9344	0	0	0	1	0	0	0	0
5	9	9344	0	0	0	0	0	1	0	0



En la parte de abajo de la figura de la página anterior se ha representado la parte de la figura anterior correspondiente a los puntos que tienen $X = 0$. Para brevedad sólo se han indicado los 4 primeros grupos (Y comprendida entre 0 y 31); para los otros, el mecanismo es idéntico. Si se quiere determinar la posición inicial correspondiente a un determinado valor de Y, los pasos a realizar son:

- Determinación del grupo G (grupo de 8 líneas) a que pertenece. El grupo G viene dado por la parte entera del cociente $Y/8$, puesto que cada grupo contiene 8 valores de Y.
- Determinación de la dirección inicial del grupo G al que pertenece Y. La posición inicial de cada grupo varía con paso 128 y, por tanto, la dirección del grupo G viene dada por $8192 + G * 128$.
- Determinación de la posición de Y dentro del grupo G.

Cada grupo empieza con un determinado valor de Y (0, 8, 16, etc.) y termina con un valor + 7 con respecto al valor inicial (contiene 8 líneas, de 0 a 7). Una línea general Y ocupa, en el interior del grupo, la posición

$$Y - G * 8$$

Efectivamente, el grupo de partida es G, y el producto $G * 8$ indica cuántas son las líneas anteriores a la Y que pertenecen a otros grupos; la diferencia entre Y y este último valor proporciona la posición de la línea Y en el punto G. Como en el interior del grupo el paso es 1024, para hallar la posición de memoria debe multiplicarse el valor anterior (posición de Y en el grupo G) para 1024 y sumar la dirección inicial del grupo G, calculado en el punto B. La segunda parte de la pantalla, cuya memoria empieza en la posición 8232, puede tratarse de manera idéntica, con la única advertencia de sustituir este valor por el 8192 utilizado por el paso B.

TABLAS DE LAS POSICIONES DE REFERENCIA EJE Y

Primera parte

Base 8192 - Ordenada (Y) entre 0 y 63

Grupo	Valores de Y		Memoria de inicio grupo
	de	a	
0	0	7	8192
1	8	15	8320
2	16	23	8448
3	24	31	8576
4	32	39	8704
5	40	47	8832
6	48	55	8960
7	56	63	9088

Algoritmo: Dirección de memoria = $8192 + 128 * \text{Grupo}$
Grupo = $\text{INT}(Y/8)$

Segunda parte

Base 8232 - Ordenada (Y) entre 64 y 127

Grupo	Valores de Y		Memoria de inicio grupo
	de	a	
8	64	71	8232
9	72	79	8360
10	80	87	8488
11	88	95	8616
12	96	103	8744
13	104	111	8872
14	112	119	9000
15	120	127	9128

Algoritmo: Dirección de memoria = $8232 + 128 * (\text{Grupo} - 8)$
Grupo = $\text{INT}(Y/8)$

En la figura de la página anterior se han representado dos tablas, una para cada una de las dos zonas, que proporcionan, en función del valor de Y, el grupo de pertenencia y la correspondiente posición inicial. Por ejemplo, el valor Y = 35 pertenece al grupo 4, que empieza en la posición 8704 y corresponde a la posición

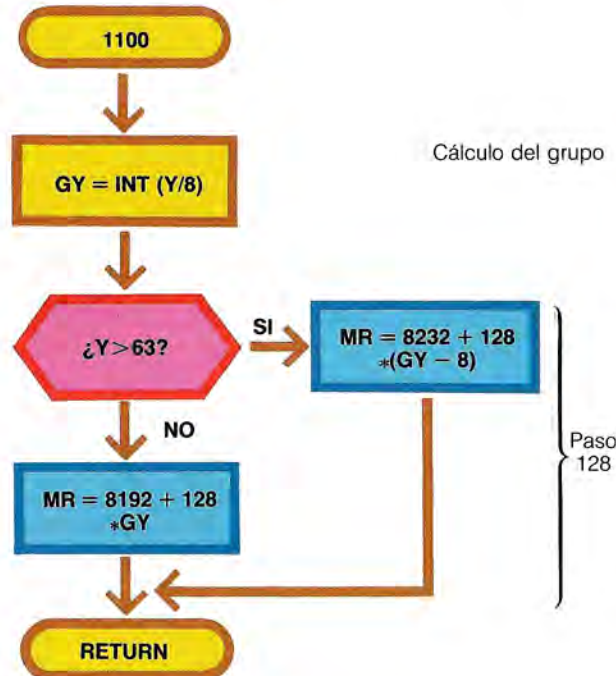
$$8704 + 1024 * (35 - 4 * 8) = 8704 + 1024 * 3 = 11776$$

Abajo se han representado los diagramas de flujo de dos subrutinas que de un valor dado de Y de la ordenada hallan la dirección correspondiente (siempre con X = 0).

DIAGRAMAS DE FLUJO PARA EL CALCULO DIRECCIONES EJE Y

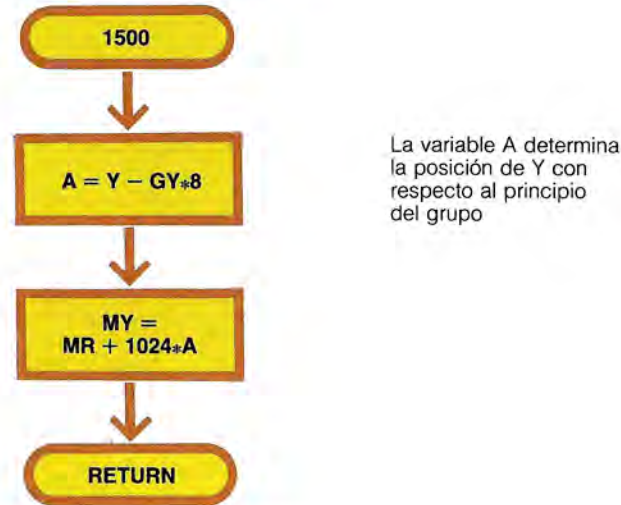
Cálculo de la dirección de referencia en el eje Y

Entrada: Y = Valor de la ordenada
Salida: MR = Dirección de referencia del grupo
GY = Grupo



Cálculo de la dirección correspondiente a la ordenada

Entradas: Y = Ordenada
MR = Dirección de referencia grupo
GY = Grupo
Salida: MY = Dirección de memoria correspondiente a Y



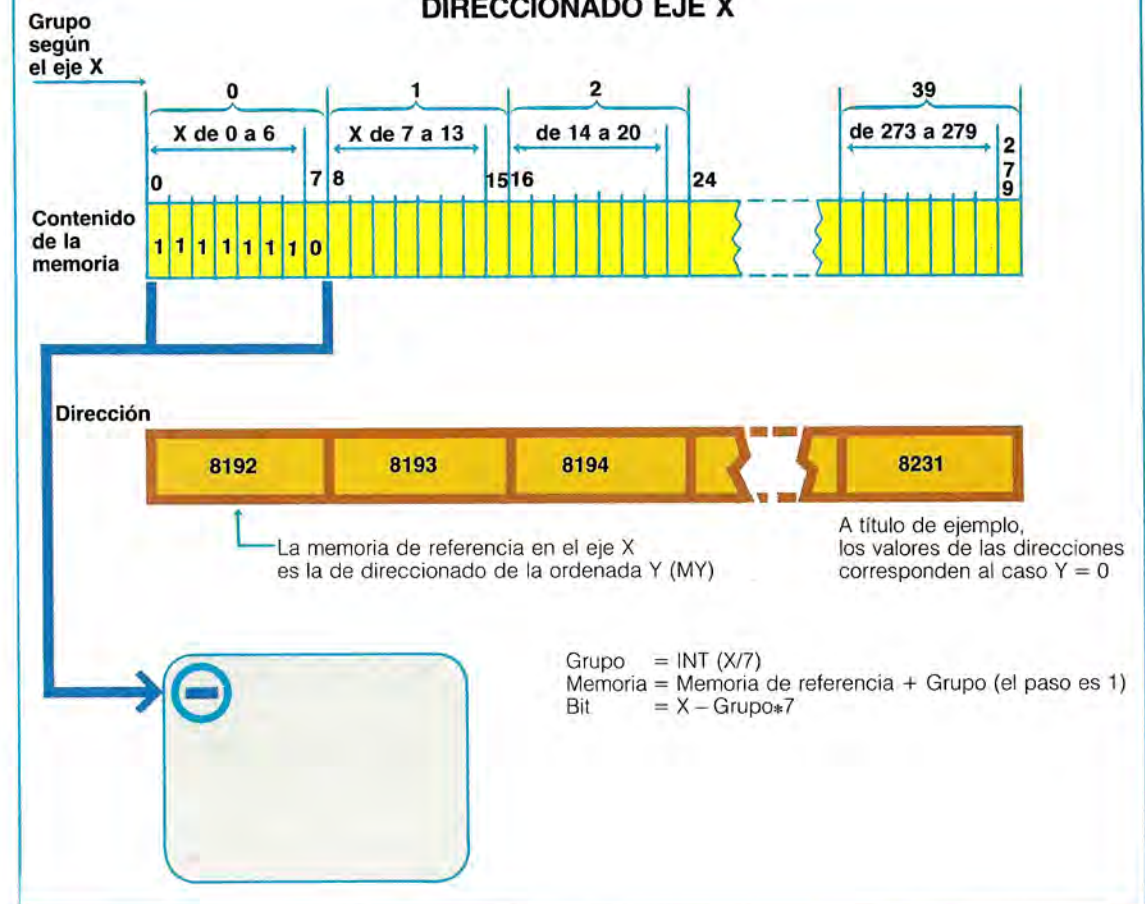
Además, en la primera (subrutina 1100) se ha implantado automáticamente el valor de referencia (8192 o 8232) en función del valor de Y. Las operaciones a realizar para determinar la dirección de memoria son muy sencillas, y el método seguido no es el más compacto. Se ha elegido voluntariamente un método muy fraccionado para permitir que el usuario intervenga, modificando los valores numéricos, para poder emplear esta técnica también en otras máquinas.

Direccionado según el eje X. Para cualquier línea (valor de Y), las diferentes columnas de la pantalla (de 0 a 39) corresponden a memorias contiguas. Por tanto, conociendo el valor inicial de la línea general Y y a la columna 0 (o mejor, a la abscisa X = 0), las otras direcciones se obtienen secuencialmente sumando 1. Cada posición de memoria contiene 8 bits y, por tanto, puede indicar el estado de 8 puntos de pantalla. Por ejemplo, el bit 0 corresponde al punto X = 0, el bit 1 a X = 1, y así sucesivamente.

En la máquina utilizada para los ejemplos, el bit 7 (la numeración empieza por 0) no se emplea para este objeto y, por tanto, cada posición de memoria contiene el estado (ON/OFF) de 7 puntos (bits 0 a 6). En el gráfico de abajo se han representado las direcciones de los puntos de pantalla que pertenecen a la línea Y = 0. La dirección de partida de la línea es 8192, y las direcciones siguientes se obtienen sumando 1 hasta un máximo de 39, que constituye el número de columnas previstas (las columnas son 40, del número 0 al número 39). Cada columna está dividida en 7 puntos de pantalla para un total de $7 \times 40 = 280$ puntos (de X = 0 a X = 279). Para direccionar un punto de abscisa X, antes debe determinarse a qué grupo pertenece (columna), calcular la dirección de memoria de este valor y, finalmente, determinar la posición del bit en la memoria. Por ejemplo, el punto de abscisa X = 16 pertenece al grupo 2, porque:

$$\text{Grupo} = \text{INT}(X/7) = 2$$

DIRECCIONADO EJE X



Por tanto, la dirección de memoria es la del principio de línea (8192) más 2, o sea

$$\text{Memoria} = 8192 + 2 = 8194$$

En la posición de memoria así identificada, el primer bit (número 0) representa el punto de pantalla que tiene $X = 14$; como el punto examinado tiene $X = 16$, estará representado por el tercer bit, o sea por el número 2 (bit 0 = punto 14, bit 1 = punto 15, bit 2 = punto 16). En general, para el cálculo del bit se tiene:

$$\text{Número del bit} = X - \text{Grupo} * 7$$

En los gráficos de abajo y de la página siguiente se han representado las dos subrutinas que realizan estos procesos. En la primera (1800) debe proporcionarse la dirección de referencia de la línea (determinada con el valor de Y) y el valor de la abscisa C. En la salida se obtiene la posi-

ción de memoria MX, que contiene el estado del punto identificado por las coordenadas X,Y, y su posición en la posición de memoria.

La segunda subrutina (1900) sirve para activar el punto de pantalla escribiendo el valor 1 en el bit correspondiente. La activación del bit debe realizarse con el operador OR para conservar la situación anterior, puesto que de otro modo se tendría la activación del bit especificado pero la desactivación de los activados anteriormente. La subrutina 1900 puede utilizarse también para leer el estado de un punto. En este caso, sólo contendrá la instrucción PEEK.

El uso descrito de la memoria, que se mostrará con más detalle más adelante, es muy frecuente en los programas de animación de figuras. Antes de desplazar una determinada figura se controla el estado de la pantalla en el punto de llegada; la respuesta de la rutina indica si el punto de llegada ya está activado o no.

Conociendo este dato, el programa de aplica-

DIAGRAMA DE FLUJO DE DIRECCIONADO DE ABCISAS

Entradas: MY = Dirección de referencia calculada por las subrutinas 1100 y 1500
X = Valor de la abscisa

Salida: MX = Dirección de la memoria eje X
BI = Número del bit a activar

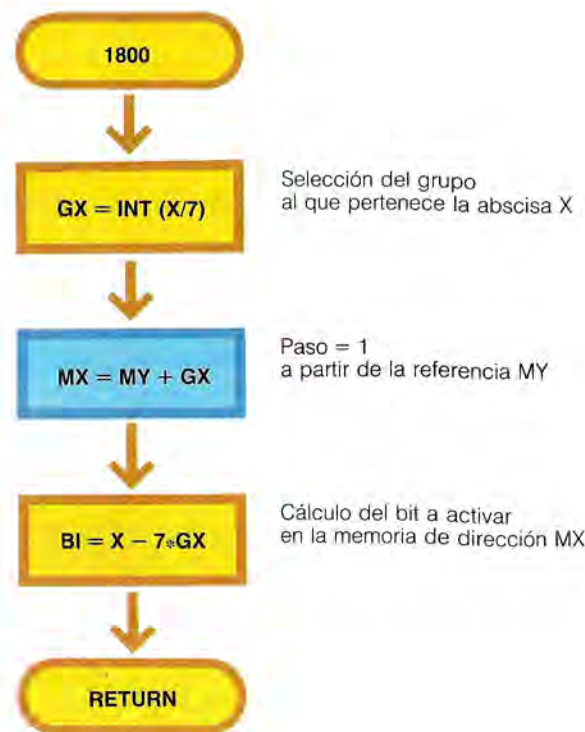
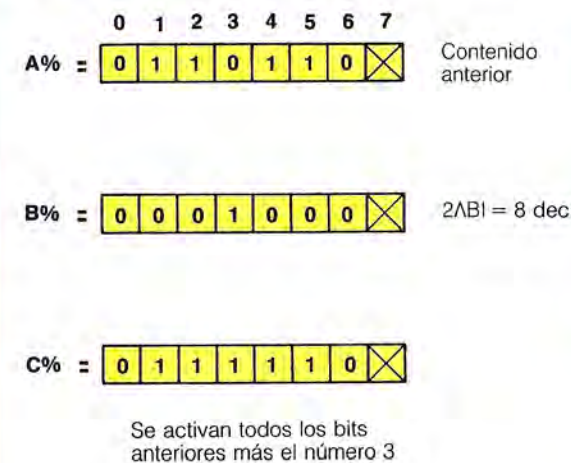


DIAGRAMA DE FLUJO DE ACTIVACION DE UN BIT

Entradas: MX = Dirección de memoria
BI = Número del bit

Ejemplo: BI = 3



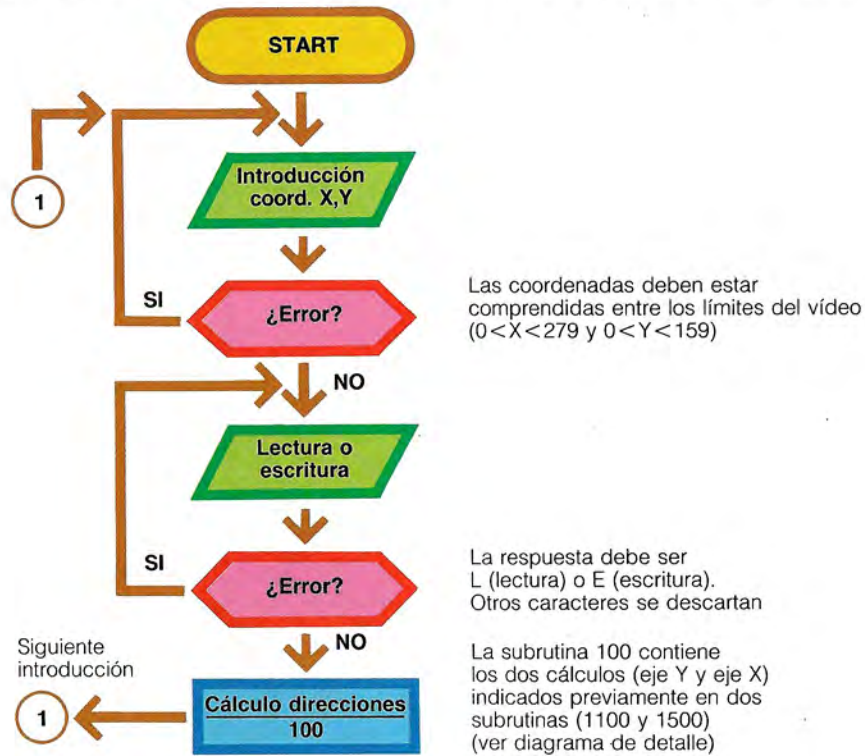
ción puede seleccionar la operación más oportuna, por ejemplo refutando el desplazamiento. En los gráficos de las páginas 1540 y 1541 se ha representado el diagrama de flujo de un programa de demostración que utiliza esta técnica. En el listado de la página 1542 se han omitido las subrutinas, puesto que ya se han presentado. La estructura en puntos de la pantalla (propia de la técnica raster) genera en algunos casos una presentación muy aproximada. En la página 1543 se ha esquematizado lo que sucede cuando se quiere dibujar un segmento inclinado con respecto a los ejes. Debido a la inclinación, algunos trozos del segmento son aproximados, con partes paralelas a uno de los ejes. Por ejemplo, en la figura de la página 1543, los segmentos 1 y 3 se representan con trazos respectivamente horizontales y verticales porque, debido a la notable inclinación, intervienen varios bits de la misma memoria. Y viceversa, con

inclinaciones próximas a 45°, para cada punto de pantalla que interviene se tiene una línea diferente y una columna diferente, con un resultado mucho mejor.

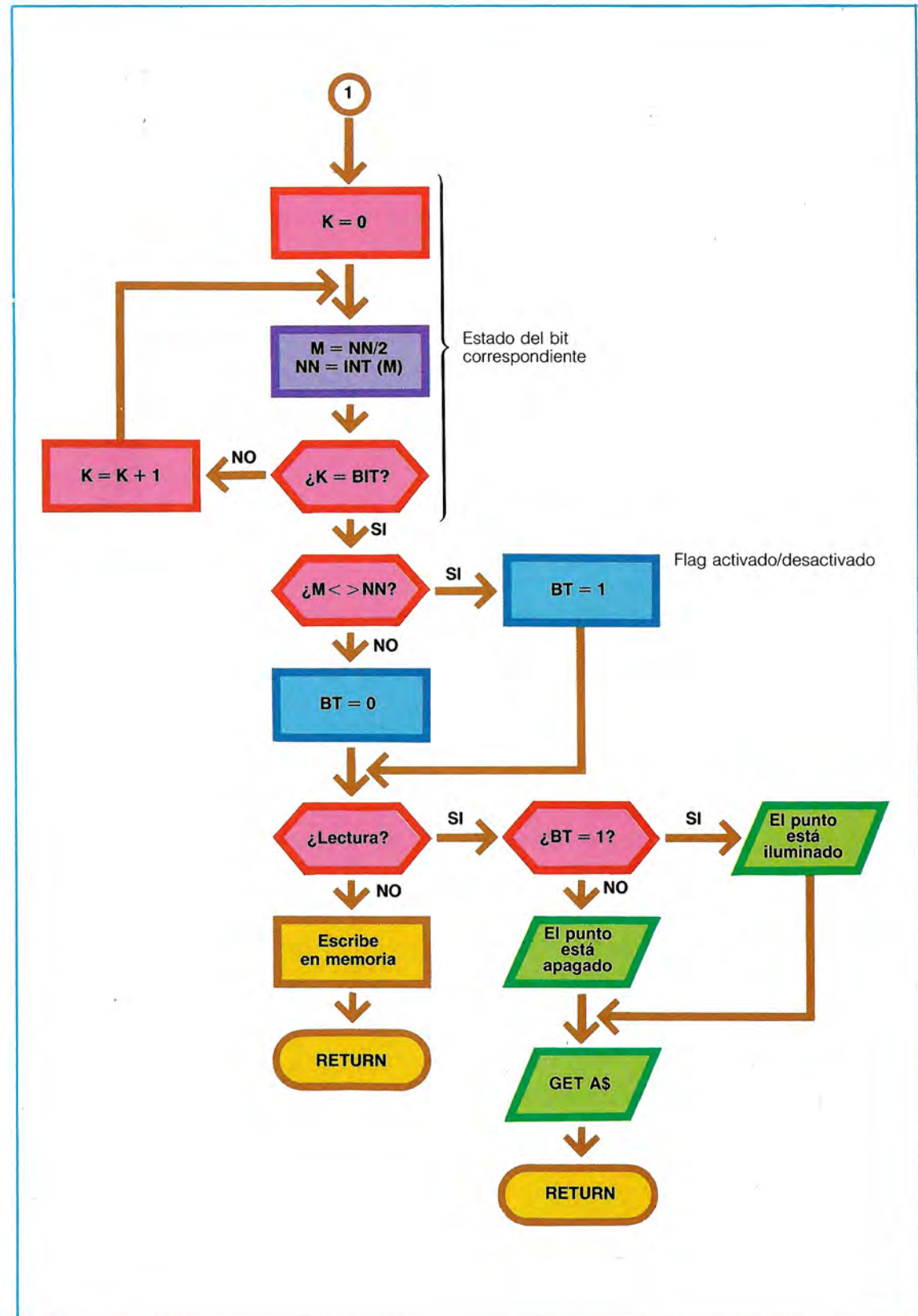
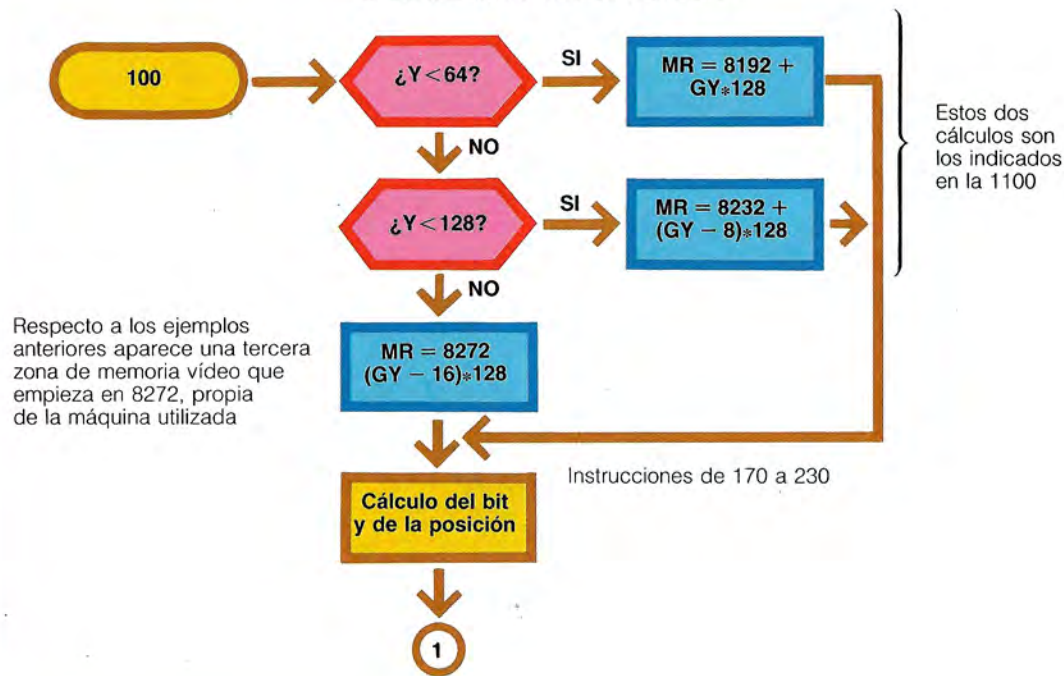
Aplicaciones del direccionado directo de la memoria vídeo

La preparación (o el reconocimiento) de una figura utilizando el direccionado a la memoria vídeo ofrece mayores posibilidades que el uso de instrucciones de alto nivel. Estas últimas están estructuradas para satisfacer las necesidades medias de un usuario que quiera generar figuras, pero no prevén funciones particulares, como por ejemplo la posibilidad de analizar la pantalla para detectar posibles figuras presentes. Para estos tipos de función es indispensable utilizar el direccionado de la memoria vídeo, para leer o modificar su contenido.

PROGRAMA DEMOSTRATIVO DE DIRECCIONADO MEMORIA GRAFICA



SUBROUTINA DE CALCULO



La posibilidad de detectar una figura presente en la pantalla es mucho más útil de lo que puede parecer a primera vista, sobre todo en dos situaciones: cuando se generan dibujos con un menú gráfico o cuando se procesan imágenes producidas con medios exteriores al ordenador (tomas con telecámara convertidas en señales digitales). La generación de un dibujo con un menú gráfico se realiza desplazando el cursor por el vídeo mediante teclas u otros dispositivos de entrada (paddle, mesa gráfica, etc.). De esta manera pueden crearse figuras sin otras limitaciones que las dimensiones y la resolución de la pantalla.

Una vez generada una figura debe memorizarse, y esto puede realizarse con varios métodos. El más directo consiste en explorar la memoria vídeo y conservar lo que hay escrito en ella. Así se obtiene una imagen numérica del dibujo que podrá someterse a otros procesos, por ejemplo cambios de escala y rotaciones, y que puede transferirse al disco. Un tratamiento análogo puede aplicarse a las imágenes tomadas con medios externos, por ejemplo con medios televisivos. La imagen original se toma de una telecámara, se digitaliza y se presenta en el vídeo. El usuario puede aportarle las modificaciones que desee introducir y después memorizarla.

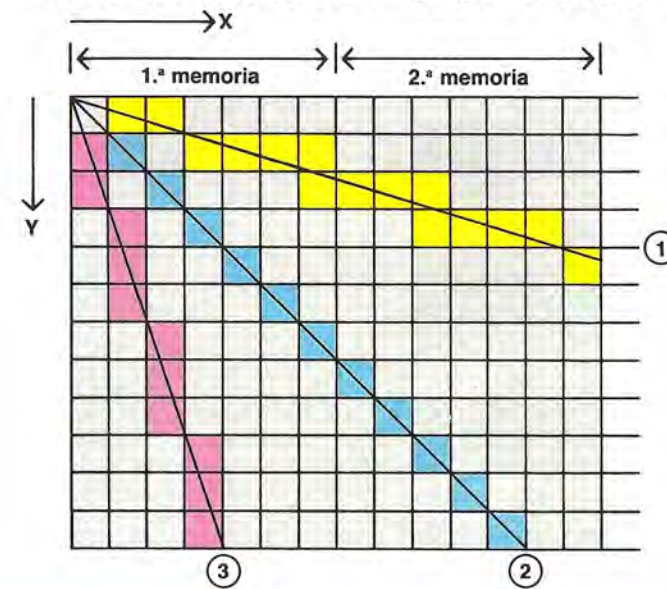
DIRECCIONADO MEMORIA GRAFICA

```

10 HGR
20 HOME
30 VTAB 23
40 INPUT "INSERTE LAS COORDENADAS: ";X,Y
50 IF X<0 OR X>279 OR Y<0 OR Y>159 THEN 20
60 INPUT "LECTURA O ESCRITURA (L/E)? ";MD#
70 IF MD#<>"L" AND MD#<>"E" THEN 20
80 GOSUB 100
90 GOTO 20
100 REM -----
110 REM Calcula direcciones
120 REM -----
130 GY=INT(Y/8)
140 IF Y<64 THEN MR=8192+GY*128: GOTO 170
150 IF Y<128 THEN MR=8232+(GY-8)*128: GOTO 170
160 MR=8272+(GY-16)*128
170 A=Y-GY*8
180 MY=MR+A*1024
190 GX=INT(X/7)
200 MX=MY+GX
210 BIT=X-7*GX
220 AX=PEEK(MX)
230 BX=2^BIT
240 NN=AX
250 FOR K=0 TO BIT
260 M=NN/2
270 NN=INT(M)
280 NEXT
290 IF M<NN THEN BT=1 GOTO 310: REM El bit está activado
300 BT=0: REM El bit está desactivado
310 IF MD#="L" THEN 370
320 REM Escritura
330 IF BT=1 THEN 360
340 CX=AX+BX
350 POKE MX,CX
360 RETURN
370 REM Lectura
380 HOME
390 VTAB 23
400 IF BT=1 THEN PRINT "EL PUNTO ESTA ILUMINADO", GOTO 420
410 PRINT "EL PUNTO ESTA APAGADO"
420 VTAB 24
430 PRINT "PULSE UNA TECLA PARA CONTINUAR ";: GET A#
440 RETURN

```

EFFECTOS DE LA RESOLUCION DE LA PANTALLA VIDEO SOBRE LA REPRESENTACION DE SEGMENTOS NO PARALELOS A LOS EJES



El direccionado mínimo es 1 bit, al que corresponde un punto luminoso. En la figura se han trazado tres líneas inclinadas. Sólo para una de ellas (45 grados) la figura representada en la pantalla aproxima bien la forma. Para las otras se obtiene una serie de segmentos paralelos. En la figura, cada punto de pantalla está representado por un cuadrado

Estas posibilidades, asociadas al uso de una pantalla en colores, han abierto nuevas aplicaciones a los ordenadores gráficos. Son muy frecuentes sus empleos en el campo de la publicidad, en el que algunas figuras reales, digitalizadas, pueden modificarse en el ordenador para dar lugar a efectos especiales. En el campo del arte han generado una verdadera escuela de pintura en la que se utiliza el haz electrónico en lugar del pincel clásico. La principal ayuda que en estos casos puede aportar el ordenador es la extremada velocidad y facilidad de las modificaciones. Las variaciones de un color o de la forma de un detalle no precisan pérdidas de tiempo por parte del usuario ni otras operaciones que no sean impartir una orden, mientras que los medios tradicionales necesitan tiempos importantes. Otra ventaja consiste en la posibilidad de memorizar algunas formas principales que, durante la preparación del dibujo, pueden reclamarse y utilizarse para formar una composición. De esta manera es posible generar un menú gráfico para aplicaciones artísticas.

Memorización de imágenes gráficas

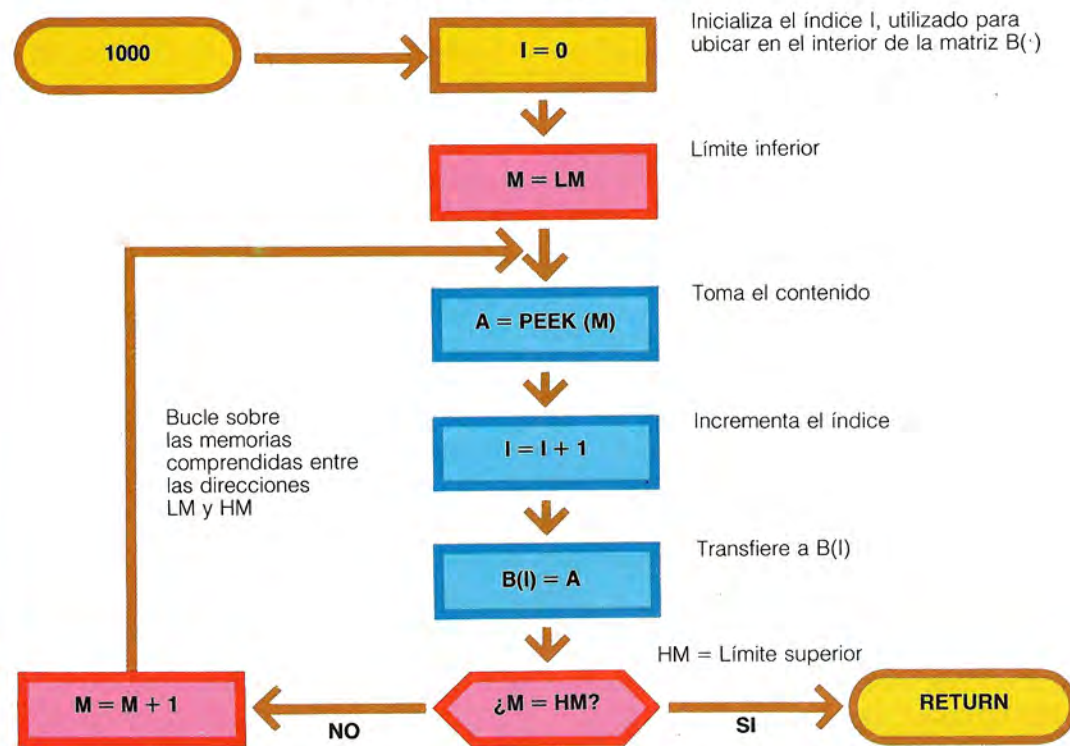
El primer paso de un proceso gráfico es necesariamente la adquisición de la figura presente en el vídeo.

Algunas máquinas tienen instrucciones idóneas

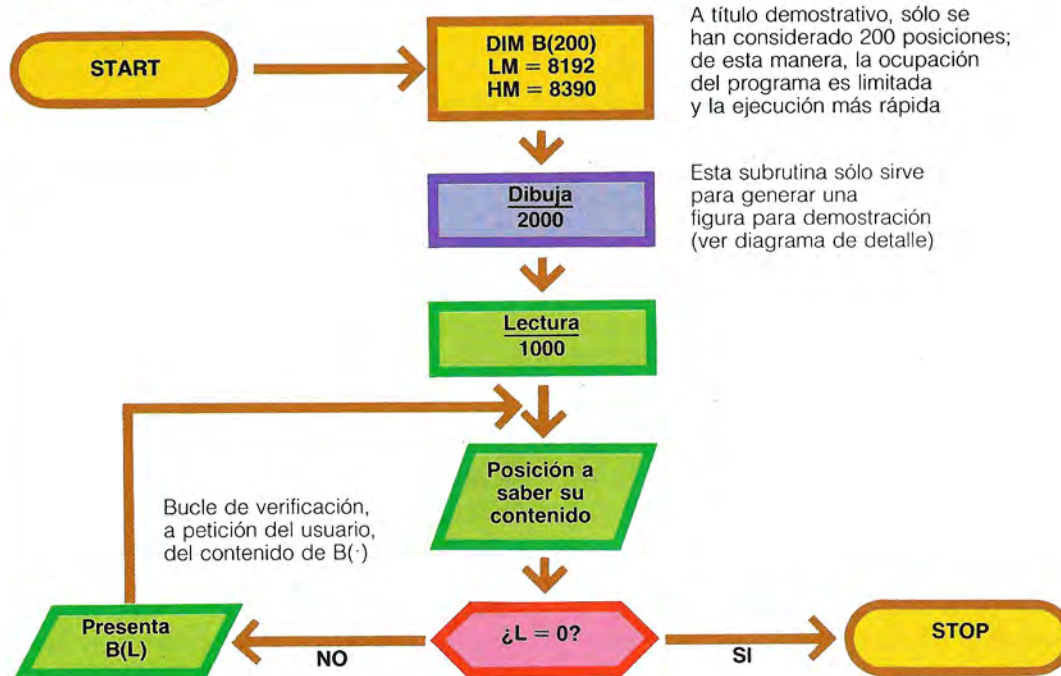
para este objeto. Por ejemplo, en la Olivetti M20, la instrucción POINT (X,Y) restituye el color del punto de coordenadas X,Y y, por tanto, su estado. Para la misma máquina existen instrucciones para memorizar una zona entera de pantalla y para representarla. Para las máquinas que no disponen de estas implantaciones Basic, debe ser el usuario el que forme las propias rutinas. Como se ha indicado, el método más sencillo y directo consiste en tomar con la instrucción PEEK el contenido de cada posición de memoria vídeo y transferirlo a una matriz definida por el usuario. La lectura de la memoria vídeo debe realizarse con la instrucción PEEK, pues no existe otra posibilidad de direccionado que la directa. Es decir, a la memoria vídeo no hay asociado ningún nombre simbólico que pueda utilizarse en el programa del usuario y, por tanto, debe direccionarse con el número de posición. La subrutina que puede realizar esta función no es otra que un bucle entre los límites máximo y mínimo de la memoria vídeo.

En la figura de la página siguiente se ha representado un diagrama de flujo parametrizado para ser adaptado a diversas posiciones de memoria y, por tanto, a diversas máquinas. En el main de prueba incluido en el listado de la rutina (pág. 1546) se utilizan las direcciones 8192 y 16383 correspondientes a la máquina emplea-

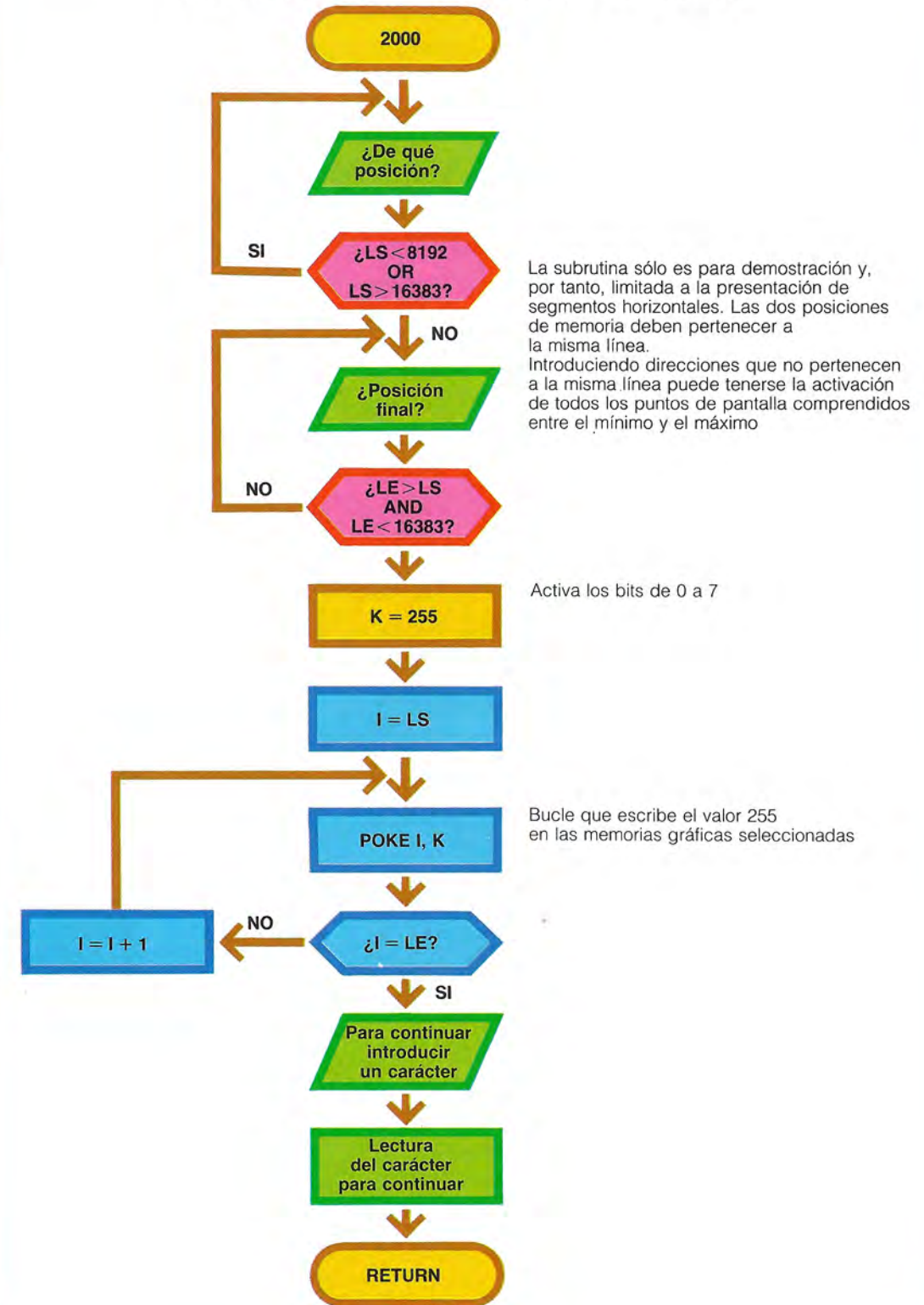
LECTURA DE LA MEMORIA VIDEO



Main de prueba



PREPARACION DE UN SEGMENTO DE PRUEBA



LECTURA Y PRESENTACION DEL CONTENIDO DE LA MEMORIA VIDEO

Versión Siprel 2010, Apple y compatibles

```

50 HGR
100 DIM B(200):LM = 8192:HM = 83
    90
110 GOSUB 2000
120 GOSUB 1000
130 HOME : VTAB 22: INPUT "INSERTAR
    POSICION ? " : L
140 IF L = 0 THEN TEXT : END
150 HOME : VTAB 22: PRINT "CONTE
    NIDO = "B(L)
155 PRINT"PULSAR UNA TECLA";: GET
    Q#
160 GOTO 130
1000 REM =====
1001 REM *LECTURA*
1002 REM =====
1003 :
1010 I = 0
1015 FOR M = LM TO HM
1020 A = PEEK (M):I = I + 1
1030 B(I) = A
1040 NEXT M
1050 RETURN
2000 REM =====
2001 REM * DIBUJA *
2002 REM =====
2003 :
2010 HOME : VTAB 22 INPUT "POSI
    CION INICIAL ?";LS
2015 IF LS (8192 OR LS) 16383 THEN
    2010
2020 INPUT "POSICION FINAL? "
    ;LE
2025 IF LE > LS AND LE <16383 THEN
    2040
2030 GOTO 2020
2040 K = 255
2045 FOR I = LS TO LE
2050 POKE I,K
2060 NEXT I
2070 HOME : VTAB 22 PRINT "PULSAR
    UNA TECLA ";:GET Q#
2080 RETURN
    
```

da (Personal Kid Siprel, mod. 2010); para otros ordenadores es necesario modificar los valores según las indicaciones del correspondiente manual. En el listado podrían aparecer otras dos instrucciones estrechamente ligadas a la máquina: HIMEM y LOMEM. Estas instrucciones se utilizan para indicar la zona que puede ser ocupada por el programa del usuario. La amplitud de la zona, por lo tanto, tiene los límites HIMEM (dirección de memoria más alta) y LOMEM (dirección de memoria más baja), que dependen de la configuración hardware y del software del sistema empleado. El área RAM total del sistema debe dividirse en varias funciones, de las que el resumen de las principales es:

- sistema operativo, por ejemplo el DOS
- funciones particulares (por ejemplo memoria vídeo de texto)
- memoria gráfica
- área usuario

En el sistema utilizado, las primeras 1024 posiciones (de 0 a 1023) son empleadas por el sistema (punteros, buffers, etc.); las de 1024 a 3071, por dos páginas de texto; sigue una zona libre (de 3072 a 8191) y después el área reservada a los gráficos (8192 a 16383).

La memoria restante debe subdividirse entre el DOS y los programas de usuario. El DOS siempre está colocado en la parte más alta y, por

tanto, para el programa del usuario queda disponible la zona entre el final de la página gráfica y el principio del DOS. La situación se muestra en la figura de abajo. Además, en este sistema es posible utilizar una segunda página gráfica (que puede gestionarse y presentarse independientemente por la primera), la cual ocupa las posiciones 16384 a 24575. El programa del usuario puede por tanto empezar a partir de la posición 24576 y debe terminar antes del principio del DOS (posición 40960).

Los dos parámetros HIMEM y LOMEM establecen estos límites para evitar que el programa entre en la zona de memoria reservada a los gráficos. En este caso particular, los dos parámetros no son necesarios, dada la limitada ocu-

pación del programa, y sólo se han mencionado con el objeto de llamar la atención sobre el problema de la ocupación de memoria. Para ser utilizado realmente para los fines de la memorización y de la recuperación de páginas gráficas, el programa mostrado anteriormente necesita algunas implantaciones. La más importante corresponde al mecanismo de memorización y, por tanto, a la amplitud del área de memoria ocupada por los buffers.

El método más conveniente consiste en la utilización de un buffer limitado a contener una sola línea cada vez (40 posiciones de memoria); para cada nueva línea, el contenido del buffer se transfiere al disco y, por tanto, el buffer queda libre para contener la siguiente línea.

DISTRIBUCION DE LA MEMORIA RAM EN EL SISTEMA SIPREL 2010

DOS	49152
	40960
Area útil para los programas de usuario	40959
	24576
2ª página gráfica	24575
	16384
1ª página gráfica	16383
	8192
Disponible	8191
	3072
Páginas de texto y gráficos en baja resolución	3071
	1024
Area de sistema	1023
	0

La anatomía del ordenador personal

Muy a menudo, en el usuario de un ordenador personal, el profundo conocimiento del lenguaje de programación no coincide con un conocimiento paralelo de la estructura hardware que utiliza. A menudo, el chasis del ordenador se contempla como una barrera impenetrable, más allá de la cual hay densos mazos de hilos que conectan misteriosas tarjetas.

La actitud de embarazo hacia los componentes hardware que se encuentra muy extendida entre los usuarios corre el riesgo de tener desagradables consecuencias, puesto que la aparición de un fallo o de un mal funcionamiento del sistema sorprende sin ninguna preparación al usuario. Quien se haya encontrado con la necesidad de hacer reparar un fallo de hardware de su ordenador personal, no habrá dejado de experimentar la escasa diligencia con que emiten sus diagnósticos los centros de asistencia, que casi siempre se traducen en largos plazos de reparación.

Sin embargo, lo que debe quedar claro de una vez por todas es que la estructura interna de un personal es muy sencilla, mucho más que los circuitos de un televisor o de un aparato de radio, los cuales procesan tanto señales analógicas como digitales.

El valiente usuario que un día decide quitar la máscara a los circuitos de los que es el legítimo propietario, invariablemente queda sorprendido por su reducidísimo tamaño y la modularidad del sistema. Las diversas funciones propias del aparato las realizan componentes (tarjetas) bien separados y fácilmente distinguibles, cuya conexión es intuitivamente elemental.

En estas condiciones, una reparación de cualquier naturaleza no debería necesitar más de quince minutos para el diagnóstico y no más de un minuto para la sustitución del circuito integrado afectado, que raramente es caro.

Por tanto, vale la pena echar una ojeada a la estructura de nuestro personal, aunque sólo sea para darnos cuenta de una vez por todas de que lo que hemos afirmado es cierto.

El sistema Siprel 2040 (Personal Kid) se presta muy bien para un estudio de la anatomía del ordenador personal. Se trata de una máquina en la que el monitor, las unidades de disco y la pantalla conviven en una sola unidad física, a la cual hay conectado un teclado separado.

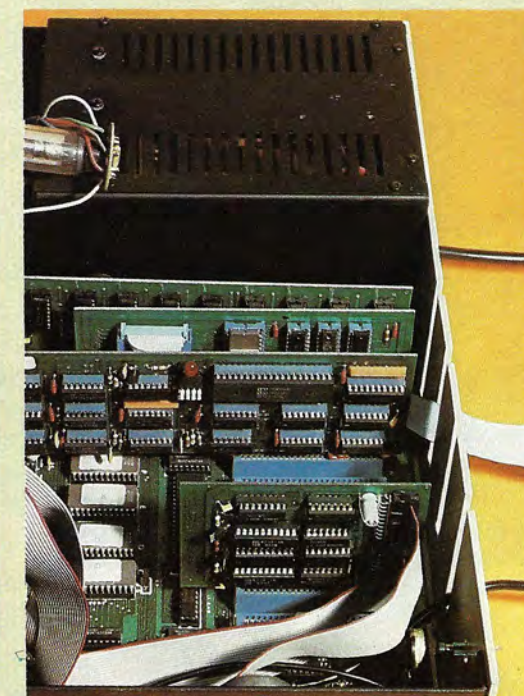
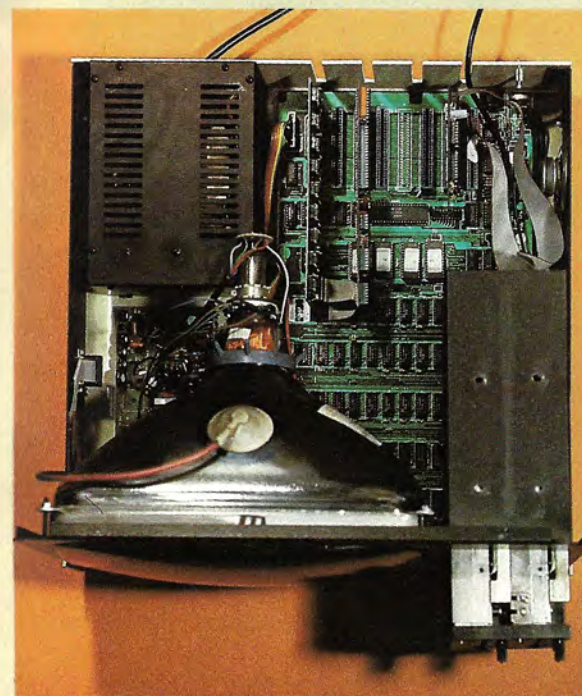
Aunque del mismo sistema hay versiones en las que las diferentes unidades están separadas, examinaremos esta configuración, precisamente porque parece la más impenetrable.

En la fotografía de abajo hemos retirado la cubierta del chasis y hemos extraído parcialmente las dos unidades de disco de su alojamiento: para ello ha bastado retirar ocho tornillos. En la fotografía de arriba a la izquierda de la página siguiente puede verse el sistema por encima. En la parte baja de la fotografía, además de las unidades de disco de tipo «slim», puede verse claramente el tubo de rayos catódicos, y en el ángulo superior izquierdo la caja del alimentador. En el fondo se entrevé la tarjeta madre, en la que hay los conectores para la inserción de las diversas tarjetas funcionales dedicadas (visibles en la parte superior de la foto).

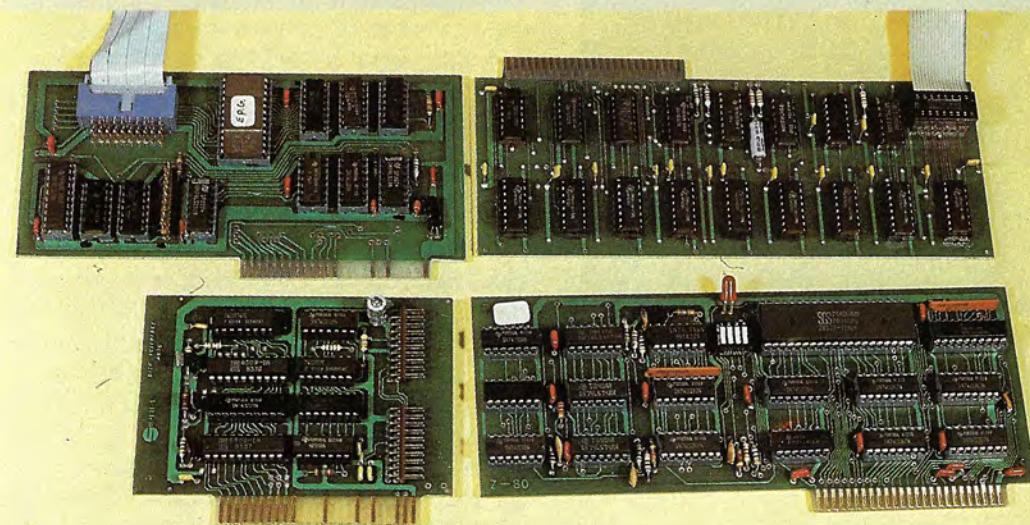
En la fotografía de arriba a la derecha de la página siguiente pueden verse las tarjetas funcionales insertadas en los correspondientes conectores. En primer plano hay la tarjeta de interfaz con las unidades de disco, de las cuales salen los dos cables planos (uno para cada unidad de disco). En segundo plano se ve la tarjeta que contiene la CPU alternativa (Zilog Z80); le



Il. Dagherroipo



Il. Dagherroipo



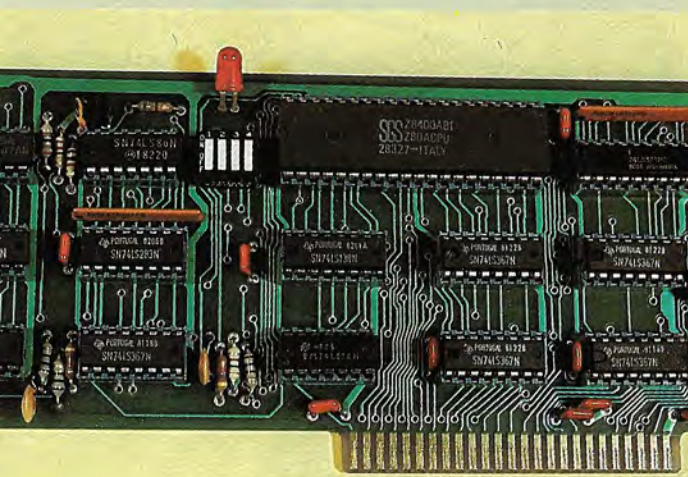
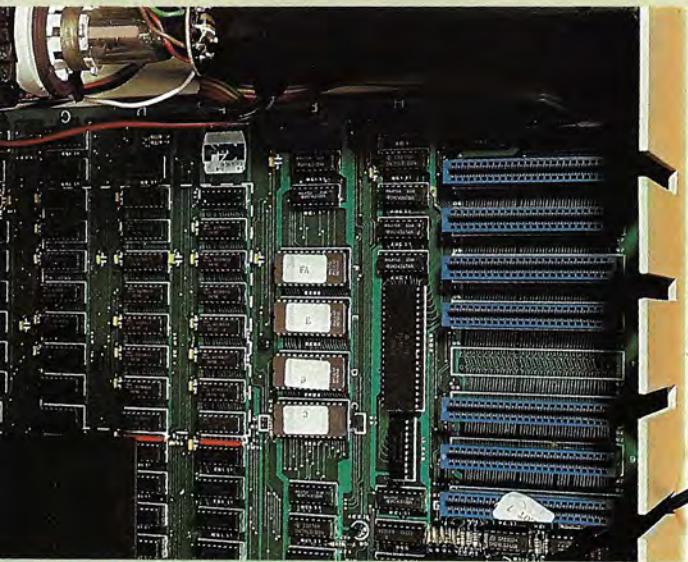
Il. Dagherroipo

sigue la tarjeta de interfaz con el vídeo y una tarjeta de expansión de la memoria RAM.

Las mismas tarjetas, extraídas de los conectores, pueden verse en la fotografía de arriba: en la parte superior izquierda, la tarjeta de gestión del vídeo de 80 columnas, a su lado la expansión de memoria, abajo a la izquierda la tarjeta de interfaz con las unidades de disco y a su lado la tarjeta con la CPU Z80.

En la fotografía de arriba de la página siguiente se ve un detalle del interior, fotografiado des-

pues de haber retirado las tarjetas funcionales de la tarjeta madre. Ahora son visibles los siete conectores y, medio oculto por el margen inferior, el zócalo para la conexión de los dispositivos externos (joystick, mesa gráfica, paddle, etc.). En este caso, al zócalo hay conectada una mesa gráfica; el cable de conexión es el que sale por el lado derecho de la fotografía. El circuito integrado de grandes dimensiones dispuesto en sentido contrario a los otros es la CPU (Rockwell 6502), mientras que los circuitos inte-



grados más pequeños de la izquierda de la imagen son memorias RAM.

Los cuatro integrados cubiertos con una etiqueta son EPROM (memorias de sólo lectura programables y borrables) en las que reside el Sistema Operativo. Las etiquetas cubren las ventanas que permiten borrar el contenido de las memorias mediante la exposición a los rayos ultravioleta, y tienen la misión de impedir el borrado accidental en el caso de una exposición prolongada a la luz solar.

El sistema Siprel se basa en la CPU Rockwell 6502 (la misma que el Apple y que el Commodore 64), que trabaja bajo DOS, pero puede completarse con una CPU Zilog Z80, que trabaja normalmente bajo CP/M (foto del centro). La CPU 6502 está en la tarjeta madre, en la que puede insertarse también una tarjeta que contiene la Z80. Insertando la tarjeta opcional, todas las unidades funcionales (memorias, dispositivos I/O, etc.) pueden ser gestionadas por la nueva CPU, obteniendo así un sistema que trabaja bajo CP/M.

La conmutación de una a otra modalidad de funcionamiento es automática. A la puesta en marcha, el sistema se predispone para trabajar bajo DOS, utilizando la 6502; insertando el disquete CP/M en la unidad e introduciendo por teclado el comando DOS de carga (IN # 6), el DOS es «sustituido» por el CP/M, y el sistema pasa a ser controlado por la Z80.

Esta dualidad es muy útil a los fines de las aplicaciones prácticas, puesto que permite que la máquina ejecute todos los programas escritos bajo los dos Sistemas Operativos más difundidos actualmente, permitiendo así la más amplia elección entre el software comercial.

En la última fotografía puede verse el teclado del sistema. En la parte central se distingue el circuito integrado destinado a la gestión del dispositivo; el integrado más pequeño, próximo a la conexión del cable, sirve para aislar el integrado principal de la línea.

Creemos haber documentado suficientemente la transparencia de un sistema de proceso bien proyectado, y haber animado a los usuarios del personal a profundizar por su cuenta en el estudio de la configuración hardware. Con esta experiencia podrá madurar el conocimiento de que cada uno de nosotros es capaz de resolver por sí mismo algunos de los problemas que se presentan normalmente en el curso de la vida de un ordenador personal.

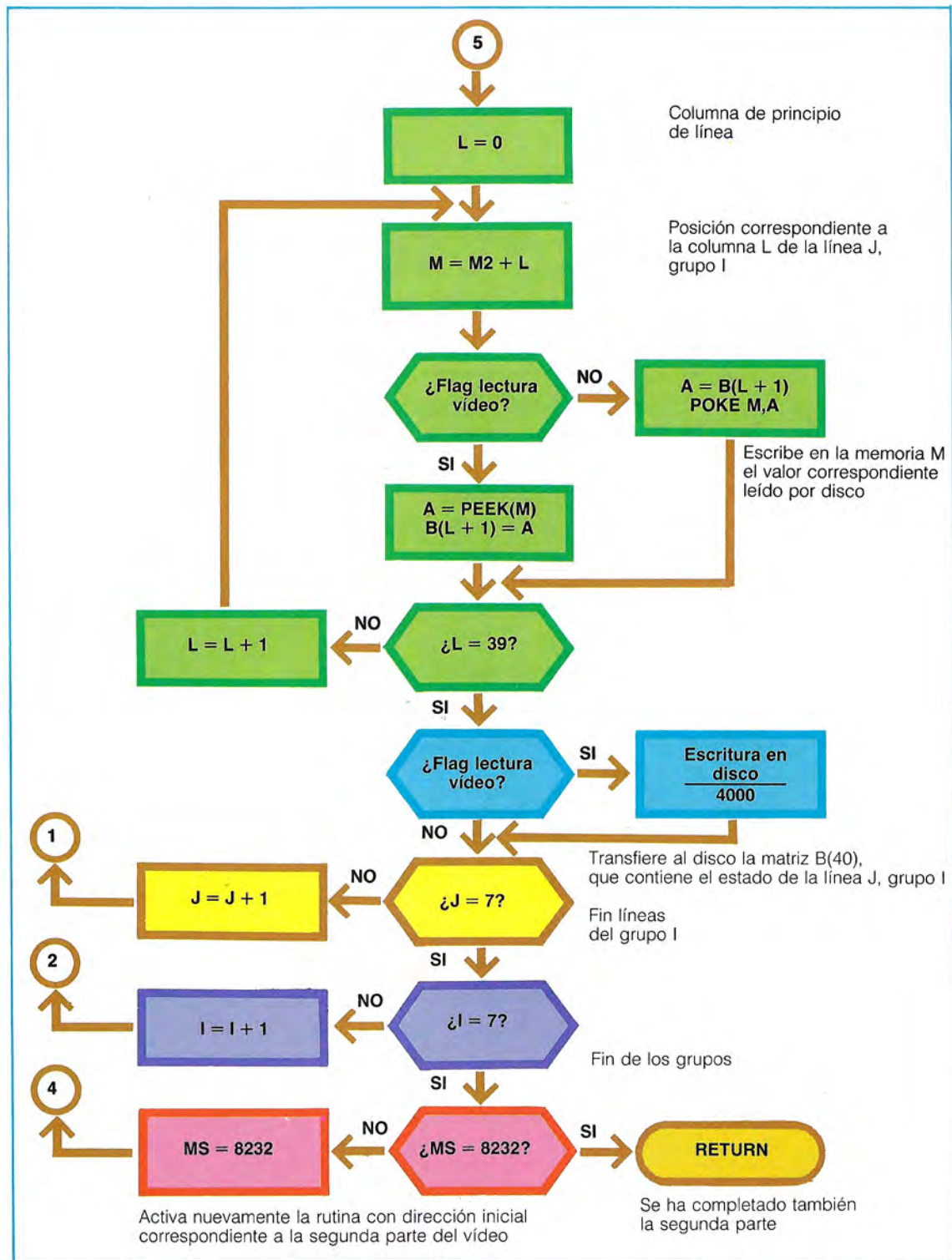
En la figura de abajo y en las páginas siguientes se muestra el diagrama de flujo de una subrutina que utiliza este método, y en las páginas 1553 y 1554 se ha representado el listado, incluido un main de prueba. Como para realizar el test del programa es necesario disponer de un dibujo en el vídeo (del que la rutina debe adquirir los puntos), en el listado se ha incluido una subrutina que permite dibujar una figura.

El método utilizado en la subrutina 1000 permite explorar línea por línea la memoria vídeo, memorizando el contenido en B; al final de cada línea, la matriz B se transfiere al disco. Si las posiciones de memoria fuesen todas contiguas, las subrutinas serían sustituidas por un simple

bucle de exploración entre la primera y la última. Y viceversa, como hay que tener en cuenta la disposición particular de las memorias gráficas, se necesitan tres bucles. El más interno explora una línea (de 0 a 39, o sea 40 memorias), mientras que los dos más exteriores son necesarios para calcular las direcciones.

Los valores indicados se refieren a la máquina utilizada y son comunes a casi todos los ordenadores compatibles Apple; por tanto, la subrutina puede ser empleada por esta familia sin ninguna modificación. La lógica empleada trata por separado las dos zonas en que está dividida la memoria interna, la primera con principio en la posición 8192, y la otra en la 8232. Una vez lla-



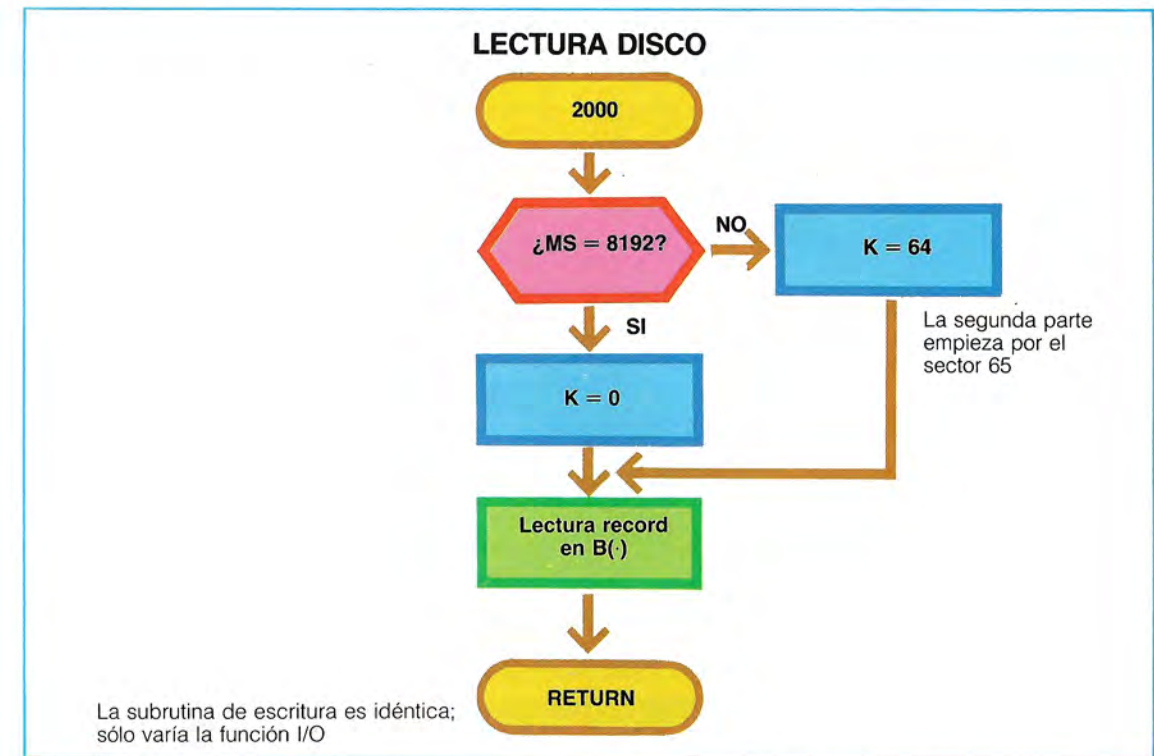


mada, la subrutina trabaja dos veces, empezando por las respectivas posiciones. Cada una de estas partes está dividida en 8 zonas de 8 líneas (de ahí la necesidad de los dos bucles con índi-

ce de 0 a 7), respectivamente con pasos de incremento 128 y 1024. El incremento justo para apuntar las zonas sucesivas se obtiene simplemente multiplicando el índice del bucle por el

paso. La lógica a seguir para la recomposición del dibujo partiendo de los datos memorizados en el disco es doble con respecto a la que se ha

utilizado por el archivado. Por tanto, la misma rutina puede emplearse para las dos operaciones; el uno o el otro modo de funcionamiento



MEMORIZACION DE IMAGENES GRAFICAS EN DISCO

```

1 REM =====
2 REM * PROGRAMA *
3 REM * DE DIBUJO *
4 REM *Y REGISTRO *
5 REM * EN DISCO *
6 REM =====
7 :
30 :
40 HIMEN: 8191
45 :
50 DIM B(40): Y = 90:X = 140:S =
55 D$ = CHR$(4)
60 PRINT D$ OPEN DIBUJO"
65 HCOLOR=3
100 HOME: VTAB 22: INPUT "LECTU
RA/ESCRITURA (L/E) ? ".A$
110 IF A$ = "L" THEN FL = 0: HGR
: GOTO 150
120 IF A$ = "E" THEN FL = 1: HRG
GOTO 140
130 IF A$ = " " THEN 200
135 GOTO 100
140 GOSUB 3000
150 GOSUB 1000: GOTO 100
200 PRINT D$ "CLOSE DIBUJO": TEXT
:END
999
1000 REM =====

```

```

1001 REM *MEMORIA VACIA *
1002 REM *****
1003 :
1010 MS = 8192
1020 FOR I = 0 TO 7
1030 M1 = MS + I * 128
1040 FOR J = 0 TO 7
1050 M2 = M1 + J * 1024
1060 IF FL = 0 THEN GOSUB 2000
1070 FOR L = 0 TO 39
1080 M = M2 + L
1090 IF FL = 0 THEN A = B(L + 1)
      : POKE M,A: GOTO 1110
1100 A = PEEK (M):R(L + 1) = A
1110 NEXT L
1120 IF FL = 1 THEN GOSUB 4000
1130 NEXT J
1135 NEXT I
1140 IF MS = 8232 THEN RETURN
1150 MS = 8232: GOTO 1020
1160 :
2000 REM *****
2001 REM * LECTURA *
2002 REM *****
2003 :
2010 IF MS < > 2192 THEN K = 64
      :GOTO 2030
2020 K = 0
2030 :
2050 PRINT D$ "READ DIBUJO"
2060 FOR P = 1 TO 40
2070 INPUT B(P)
2080 NEXT P
2150 PRINT D$: PRINT RETURN
3000 REM *****
3001 REM *DIBUJA*
3002 REM *****
3003 :
3005 H$ = "H"
3010 HCOLOR= 3
3015 H$ = "H"
3020 HOME : VTAB 22: PRINT "I=ENCIMA
      /M = DEBAJO/K = DERECHA/J = IZQ."
3025 PRINT "S=SALVA"
3030 VTAB 1 GET Q$ PRINT
3040 IF Q$ = "I" THEN Y = Y + S:
      GOTO 3200
3050 IF Q$ = "M" THEN Y = Y - S:
      GOTO 3200
3060 IF Q$ = "K" THEN X = X + S:
      GOTO 3200
3070 IF Q$ = "J" THEN X = X - S:
      GOTO 3200
3080 IF Q$ = "C" THEN 3010
3090 IF Q$ = "S" THEN PRINT RETURN

3100 GOTO 3030
3200 H$ = "H": GOTO 3030
4000 REM *****
4001 REM *ESCRITURA*
4002 REM *****
4003 :
4010 IF MS < > 8192 THEN K = 64
      : GOTO 4030
4020 K = 0
4030 :
4050 PRINT D$ "WRITE DIBUJO"
4060 FOR P = 1 TO 40
4070 PRINT B(P)
4080 NEXT P
4150 PRINT D$: PRINT : RETURN

```

está indicado por el valor de un flag activado en la llamada.

El empleo de esta rutina en otras máquinas puede necesitar modificaciones sustanciales; por ejemplo, disponiendo de una instrucción que pueda leer el estado de un punto (o el color), deben reestructurarse los bucles eliminando los más externos y conservando sólo dos, uno para las líneas y el otro para las columnas.

Cálculo de áreas y perímetros

La lectura del estado de un punto, o mejor, de la memoria que contiene su estado, puede ser indispensable en otras aplicaciones, como por ejemplo en la escritura de software para los videojuegos. En estos programas siempre hay necesidad de mover una figura comprobando las colisiones con otras figuras presentes en el vídeo. En algunas máquinas dedicadas a este uso, este control puede realizarse con una instrucción Basic, que en los personales de empleo general está ausente. Para estos últimos puede utilizarse el método de la lectura directa en memoria vídeo para determinar si la zona que deberá ser ocupada después del desplazamiento está vacía o ya contiene una figura. Es decir, si en la zona de llegada los valores no son todos cero, se tendrá una colisión entre la figura en movimiento y la existente en la zona examinada. En apariencia, este método presenta dificultades limitadas. En realidad, una rutina que

pueda realizar las funciones descritas puede ser incluso notablemente compleja.

Un problema análogo al de la colisión, pero mucho más sencillo, se presenta en el cálculo de las áreas y de los perímetros. En muchas aplicaciones gráficas, a la construcción del dibujo debe seguir una fase de proceso para hallar las dimensiones del objeto representado. Ultimado el dibujo, en la memoria gráfica del ordenador hay a escala todas las dimensiones y, por tanto, no es difícil estructurar algunas subrutinas para el cálculo de las características geométricas.

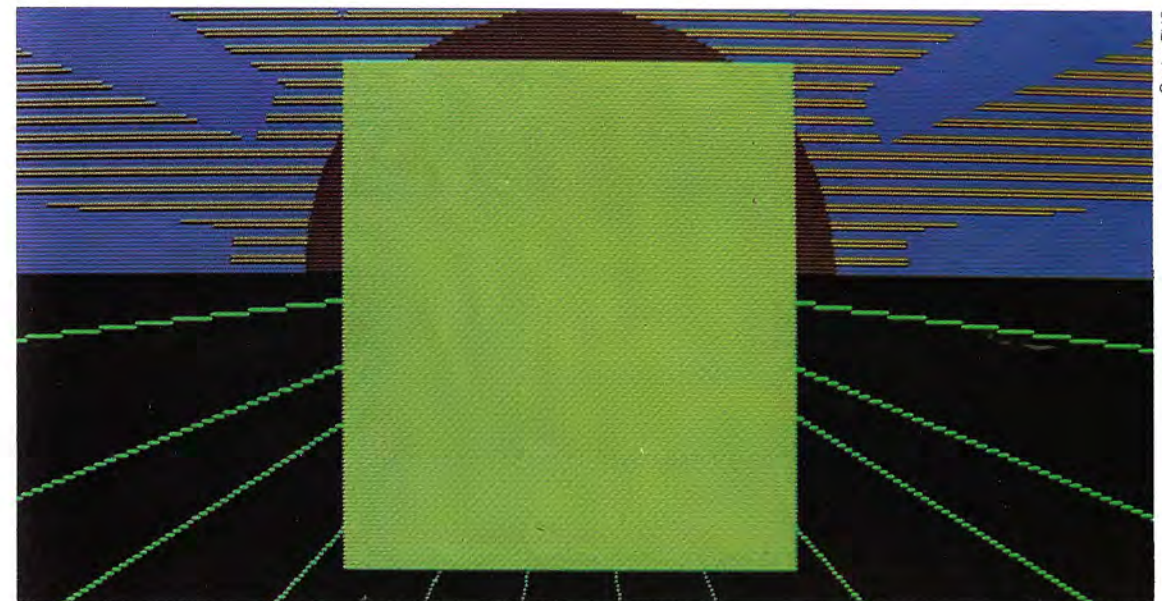
El problema puede enfocarse de dos maneras, según el método de adquisición del dibujo.

El primer método prevé que la figura, generada por el usuario con un menú gráfico, ya esté en la memoria. En este caso, debe analizarse toda la memoria vídeo tomando punto por punto el contorno y la superficie de la figura.

En el segundo, el cálculo se realiza mientras la figura se adquiere; es el caso de introducción por medio de la mesa gráfica.

Como se vio, este periférico es un transductor de posición que, momento a momento, proporciona al ordenador las coordenadas del elemento de colimación; el dibujo se adquiere desplazando esta referencia a lo largo del contorno de la figura. De esta manera, el cálculo del perímetro es inmediato y permite calcular la distancia de cada punto al que le precede. El cálculo esquematiza cada desplazamiento con un seg-

Inserción de una ventana vídeo en una imagen memorizada.



mento de recta, y es tanto más preciso cuanto más pequeños son los desplazamientos.

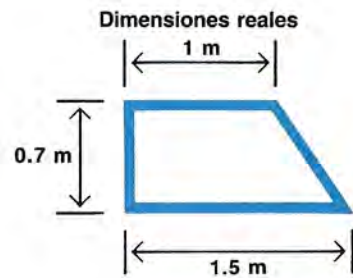
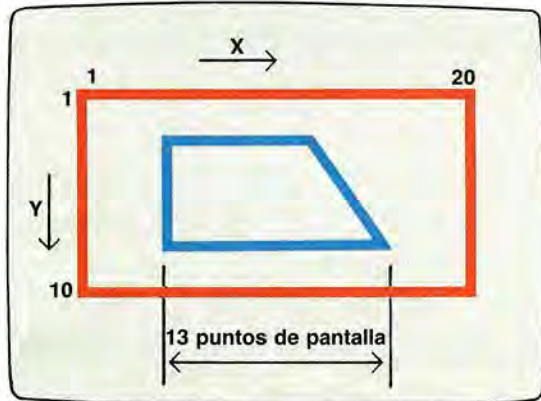
En realidad, el usuario no advierte esta segmentación; el movimiento de la referencia a lo largo del contorno es continuo, y es misión del software dividirlo en cantidades suficientemente pequeñas para asegurar una buena precisión, pero de manera que no haga demasiado lenta la ejecución del programa.

Para determinar la superficie, el dibujo debe estar terminado, y por tanto, también en este caso, el método más sencillo consiste en examinar la memoria vídeo y realizar el cálculo sobre la imagen memorizada de la figura.

En la figura de abajo se ha representado la ilustración gráfica del método. Como figura se ha elegido un trapecio, cuya área calculada matemáticamente es de 0,875 m² (las dimensiones

son en metros). Suponiendo que se ha adquirido la figura y que se ha representado con la base mayor igual a 13 puntos de pantalla, el factor de escala es 1.5/13, por lo que cada punto de pantalla vale 0,11... m y la altura en puntos de pantalla es 6,06... (altura en puntos de pantalla = 13/1.5 * 0,7). Este dimensionado es completamente ineficaz y se ha elegido sólo por motivos gráficos. En realidad, para representar este dibujo, la escala más adecuada es 100. De esta manera, la base sería de 1.5 x 100 = 150 puntos de pantalla y la altura de 70, obteniendo una mejor precisión en el cálculo. En la parte inferior de la figura de abajo se ha representado el mapa de memoria correspondiente a la imagen. Las numeraciones no son las reales de la pantalla de vídeo, sino que corresponden a una zona de la pantalla alrededor de la figura; sin

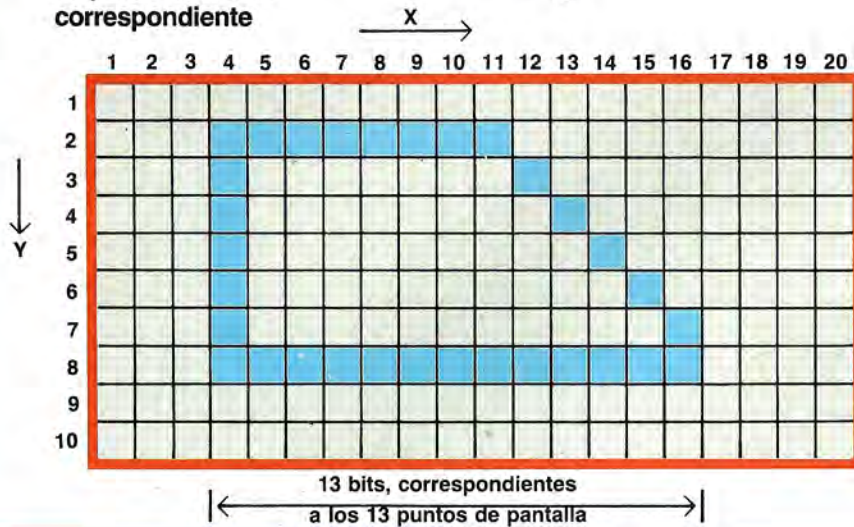
FIGURA REPRESENTADA EN LA PANTALLA



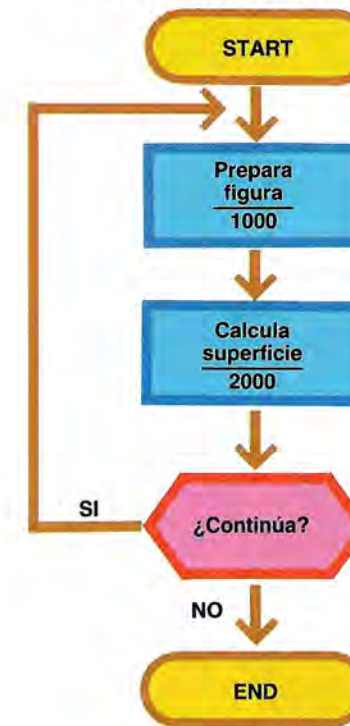
$$S = \frac{2.5}{2} \cdot 0.7 = 0.875 \text{ m}^2$$

Mapa de memoria correspondiente

$$\text{Factor de escala} = \frac{1.5}{13}$$



PROGRAMA PRINCIPAL



embargo, la ilustración del método continúa siendo válida. El cálculo de la superficie consiste en examinar el área de memoria involucrada y en determinar si cada punto es interior a la figura o no. La suma de los puntos internos proporciona, a escala, la superficie.

La exploración de la memoria puede realizarse con dos bucles, el más externo según el eje Y (en el ejemplo de 1 a 10) y el más interno según el eje X (de 1 a 20).

Por ejemplo, poniendo Y = 1 y examinando todas las posiciones de memoria que pertenecen a esta línea, no se encuentra ningún bit activo y, por tanto, en la línea Y = 1 no existen puntos del dibujo. Para Y = 2, en la columna 4 se encuentra el primer bit activo; como es el primero, se memoriza (en la variable XI) como principio de una línea interna de la figura. Prosiguiendo la exploración de la línea, cada vez que se encuentra un bit activo (después del primero), su posición se memoriza en la variable XF. Al final del bucle interno (X de 1 a 20) se tendrá XF = 11, y la diferencia XF - XI proporciona la longitud de la primera línea interna (en particular, para Y = 2, esta línea es la base menor del trapecio considerado).

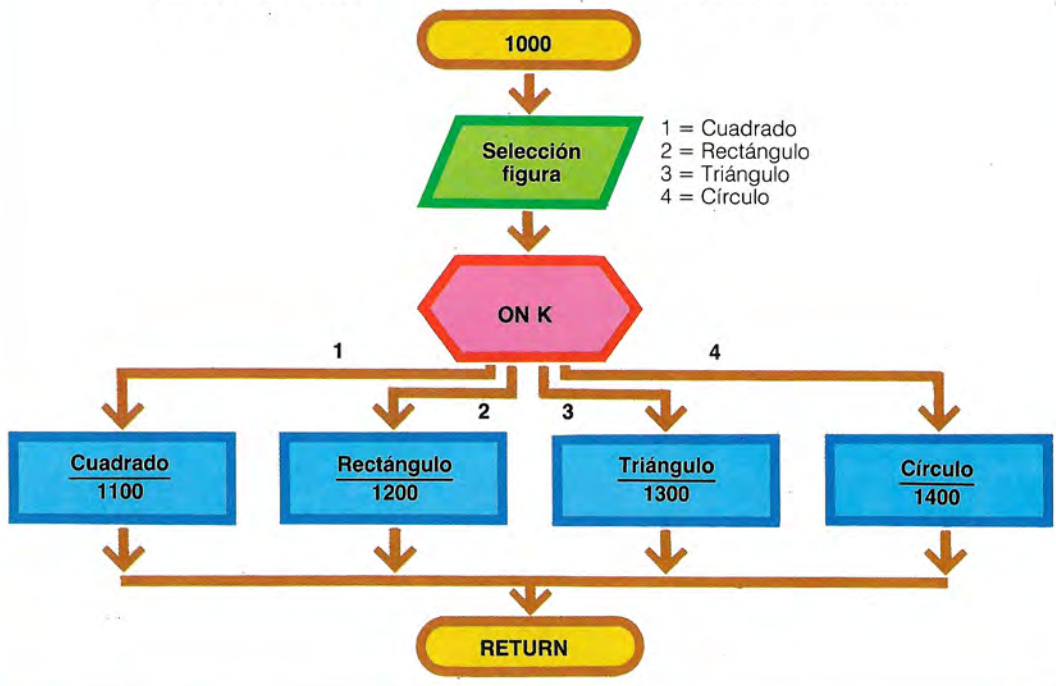
Como el dibujo tiene una escala de 1.5/13, la longitud del segmento en dimensiones reales es $(1.5/13) * (XF - XI) = (1.5/13) * 8 = 0.92... \text{ m}$. Obsérvese la diferencia que hay con la medida real de la base menor (1 m). Esto se debe al factor particular de escala adoptado. Con el valor 1.5/13, la base menor debería tener una longitud de 8.66... puntos de pantalla, pero este valor debe llevarse a 8, con el consiguiente error*. Elijiendo una escala más adecuada, este error sería despreciable. El segmento detectado de esta manera, de longitud 0.92, tiene una dimensión según el eje Y igual a 1 punto de pantalla, o sea 1.5/13 de metro, y multiplicando esta cantidad por la longitud se obtiene el valor del primer elemento de superficie.

El procedimiento continúa para todas las líneas sumando cada elemento de superficie así hallado. Al final, la suma total proporciona el área de la figura.

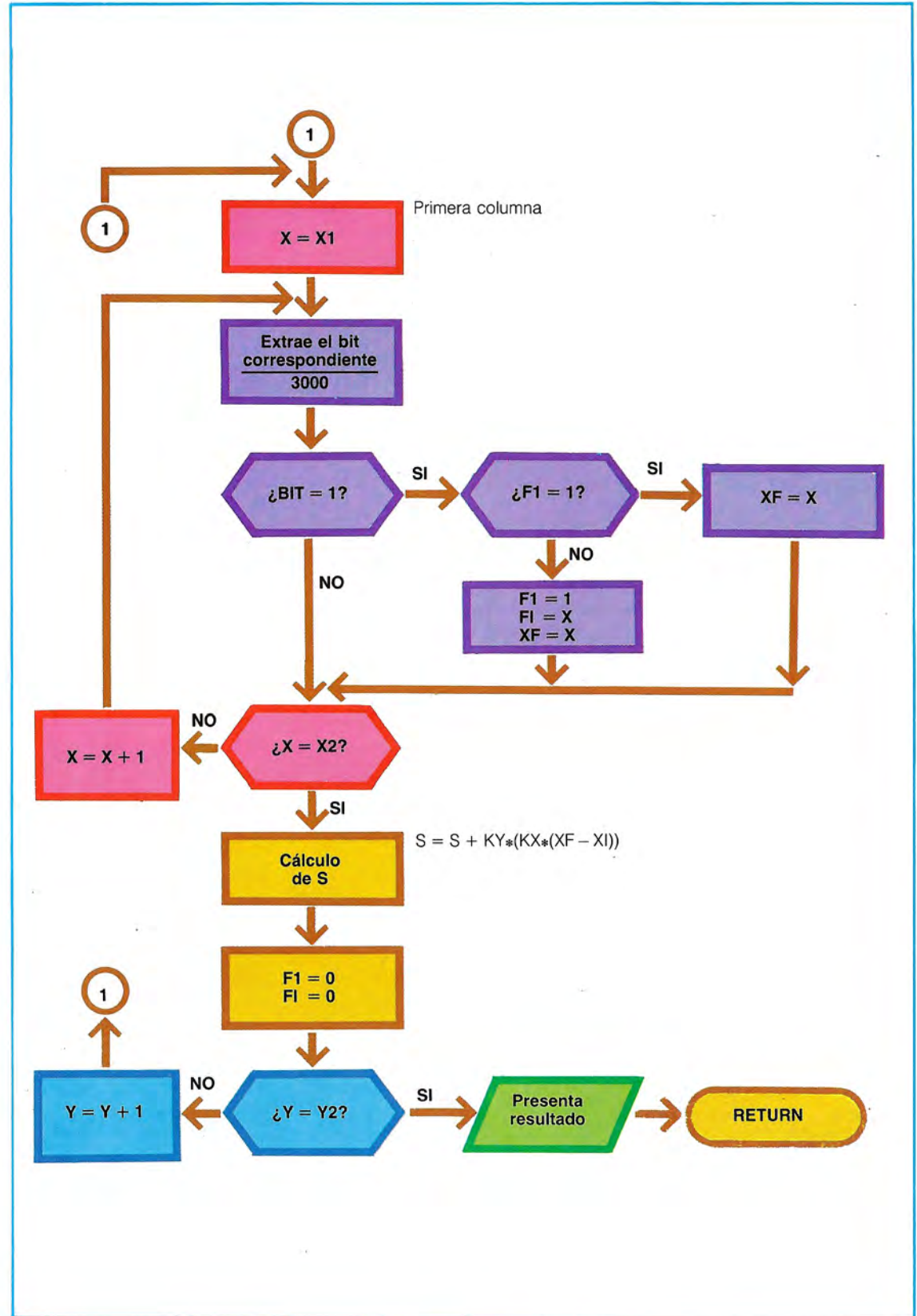
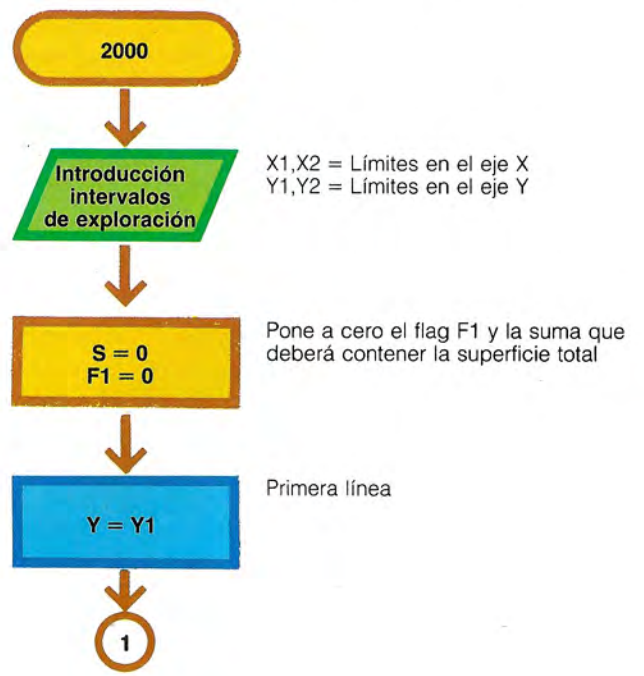
En las figuras de las páginas 1557 a 1559 se han representado los diagramas de flujo de un programa de demostración.

* La memoria vídeo y la pantalla no pueden direccionarse con valores fragmentarios.

SUBROUTINA DE PREPARACION DE LA FIGURA DE TEST



SUBROUTINA DE CALCULO DE LA SUPERFICIE



La parte inherente al problema específico es la subrutina 2000, mientras que las otras sólo sirven para preparar algunas figuras geométricas en el vídeo. Los correspondientes listados se han representado en las páginas 1561 a 1563. En el programa se ha utilizado, con algunas variaciones, la subrutina ya presentada para el cálculo de la posición de memoria dadas las coordenadas de la pantalla. Esta versión, a diferencia de la anterior, está reducida a lo esencial menos las líneas 3142 a 3150, que tienen otra función. En este caso particular, el tiempo de proceso es notable, sobre todo en Basic interpretado. Para evitar largas esperas sin que la máquina dé señales de actividad, se han introducido las líneas 3142 a 3150, que llenan el contorno de la figura a medida que se va realizando el proceso. Siempre siguiendo la misma lógica, se ha introducido el flag F2 que después puede eliminarse también en la 2071. Este flag tiene la misión de interrumpir la activación de los puntos de pantalla cuando las coordenadas son exteriores a la figura.

Finalmente debe observarse que las figuras geométricas presentadas y los correspondientes valores de las áreas calculadas no son precisas ni congruentes con el resto del programa;

el objetivo principal es el de demostración del uso de la subrutina 2000, evitando excesivas complicaciones en otros puntos. En particular, el dibujo de las figuras y las áreas halladas por el programa no corresponden a las áreas halladas analíticamente; la diferencia se debe a la forma en que se ha dibujado la figura.

Por ejemplo, para el cuadrado, introduciendo como longitud del lado el valor 10 y como origen de la figura el punto de coordenadas 10,10, el primer lado vertical se obtiene trazando un segmento entre el origen y el punto 10,20. De esta manera, la longitud en puntos de pantalla del lado es 11 y, por tanto, el área hallada tiene una diferencia del 10% con respecto a la calculable; este defecto desaparece si la figura está dibujada correctamente.

El método indicado es de tipo gráfico, puesto que el área se halla procesando gráficamente la figura. También existen algunos métodos analíticos, con los cuales el resultado (cálculo del área) se obtiene mediante la aplicación de fórmulas matemáticas. Sin embargo, estos métodos sólo pueden aplicarse en casos particulares (por ejemplo polígonos), y requieren la descripción analítica de la figura (por ejemplo, las coordenadas de los vértices); éstos corresponden

Reconstrucción gráfica de un cristal cúbico en el ordenador.

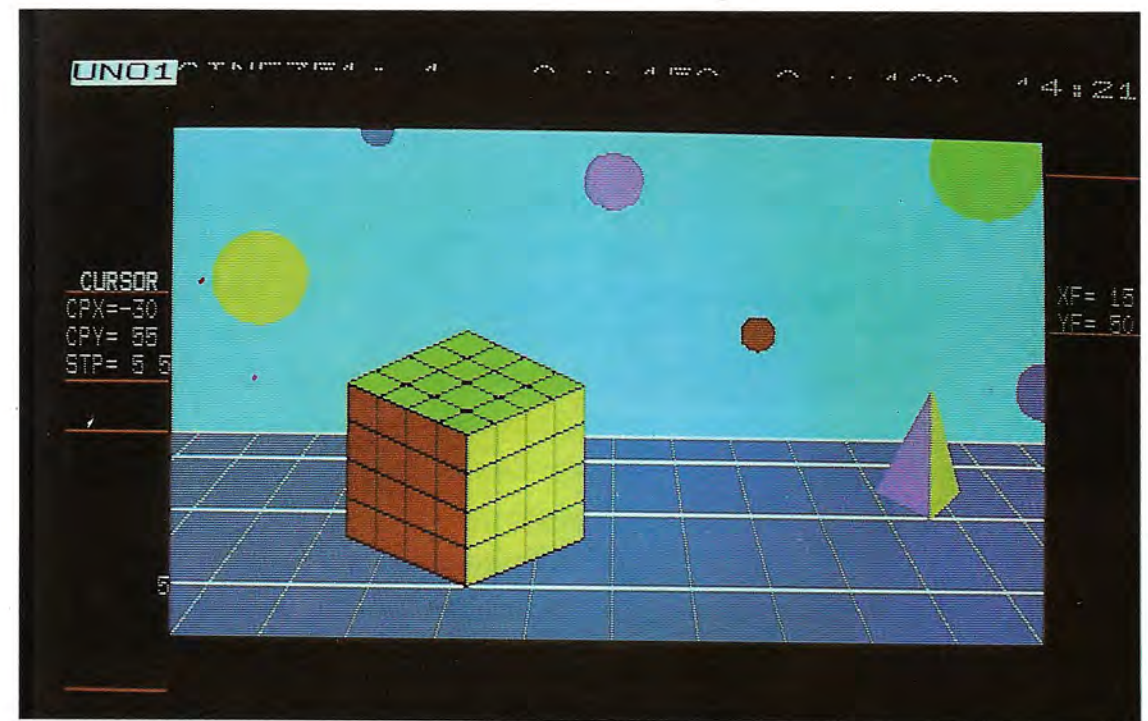
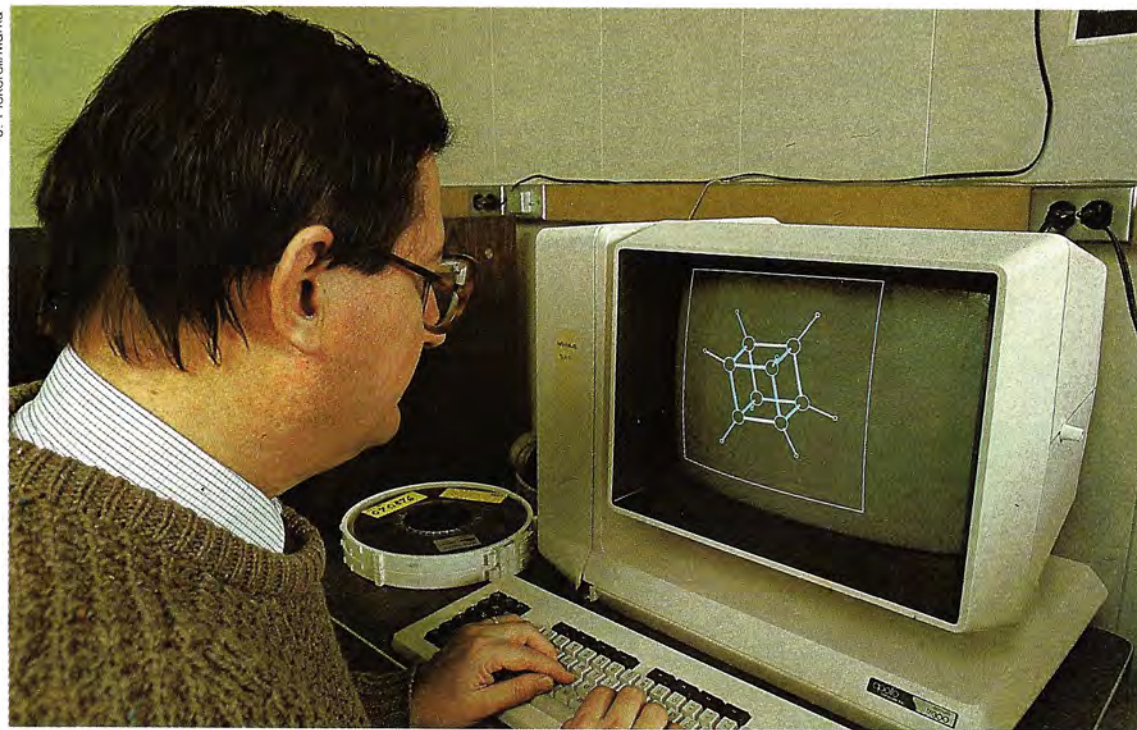


Imagen gráfica tridimensional en un monitor en colores.

PROGRAMA PARA EL CALCULO DE AREAS

Versión Siprel 2010, Apple y compatibles

```

10 REM
20 REM : FILE = AREAS
30 REM
40 KY = 1:KX = 1
50 GOSUB 1000
60 GOSUB 2000
70 REM
80 INPUT "CONTINUA ":A$
90 IF A$ = "SI" GOTO 50
92 END
1000 REM :FIGURAS DE PRUEBA
1005 REM
1010 HGR
1015 PRINT "SELECCION (1,2,3,4)
"
1020 GET R
1025 IF R < 1 OR R > 4 GOTO 1015
1030 ON R GOSUB 1100,1200,1300,1
400
1040 RETURN
1050 REM
1052 REM *****
1054 REM
1100 REM : CUADRADO
1105 REM
1110 INPUT " LADO ":L
1115 INPUT "POSICION (X,Y) ":X
0,Y0
1120 HPLOT X0,Y0 TO X0 + L,Y0 TO
X0 + L,Y0 - L, TO X0,Y0 - L TO
X0,Y0
1121 A = L * L
1122 PRINT " AREA = ":A
1125 RETURN
1130 REM
1132 REM *****
1134 REM
1200 REM : RECTANGULO
1210 INPUT " LADOS ":LX,LY
1215 INPUT " POSICION (X,Y) ":X
0,Y0
1220 HPLOT X0,Y0 TO X0 + LX,Y0 TO
X0 + LX,Y0 - LY TO X0,Y0 - L
X TO X0,Y0
1221 A = LX * LY
1222 PRINT " AREA = ":A
1225 RETURN
1230 REM
1232 REM *****
1234 REM
1300 REM : TRIANGULO
1305 REM
1310 INPUT " BASE Y ALTURA ":B,
H
1315 INPUT " POSICION (X,Y) ":X
0,Y0
1320 HPLOT X0 + B,Y0 TO X0,Y0 TO
X0,Y0 + H TO X0 + B,Y0
1321 A = B * H / 2
1322 PRINT " AREA = ":A
1325 RETURN
1330 REM
1332 REM *****

```

```

1334 REM
1400 REM : CIRCULO
1405 REM
1410 INPUT " RADIO " : R
1415 INPUT " CENTRO (X,Y) " : X0,Y
0
1520 X1 = X0 + R : Y1 = Y0
1525 FOR A = -0.05 TO 6.3 STEP
0.05
1530 X2 = X0 + R * COS (A)
1535 Y2 = Y0 + R * SEN (A)
1540 HPOINT X1,Y1 TO X2,Y2
1545 X1 = X2 : Y1 = Y2
1550 NEXT A
1555 A = 3.14 * R ^ 2
1560 PRINT " AREA = " : A
1560 RETURN
2000 REM
2005 REM : AREAS
2010 REM
2015 INPUT " LIMITES EJE X " : X1,
X2
2020 INPUT " LIMITES EJE Y " : Y1,
Y2
2025 S = 0 : F1 = 0 : XI = 0 : XF = 0
2030 FOR Y = Y1 TO Y2
2035 FOR X = X1 TO X2
2040 GOSUB 3000
2045 IF B11 = 0 GOTO 2060
2050 IF F1 = 1 THEN XF = X : GOTO
2060
2055 XI = X : XF = X : F1 = 1
2060 NEXT X
2062 PRINT " Y = " : Y, " XI = " : XI, " XF
= " : XF
2065 S = S + KY * (KX * (XF - XI)
)
2070 F1 = 0 : XI = 0 : XF = 0
2071 F2 = 0
2075 NEXT Y
2080 PRINT " AREA = " : S
2082 GET D#
2085 RETURN
2090 REM
2092 REM *****
2094 REM
3000 REM : EXTRAER EL BIT
3005 REM
3010 REM ENTRADAS :
3015 REM X,Y
3016 :
3020 REM
3025 REM SALIDAS :
3030 REM BIT=0.1
3035 REM
3036 :
3040 GY = INT (Y/8)
3045 IF Y < 64 THEN MR = 8192 +
GY * 128 : GOTO 3070
3050 IF Y < 128 THEN MR = 8232 +
(GY - 8) * 128 : GOTO 3070
3060 MR = 8272 + (GY - 16) * 128
3070 A = Y - GY * 8
3080 MY = MR + A * 1024
3090 GX = INT (X / 7)
3100 MX = MY + GX : BIT = X - 7 * G
X
3105 AX = PEEK (MX) : BX = 2 ^ BIT
3110 NN = AX
3115 FOR K = 0 TO BIT
3120 M = NN / 2
3125 NN = INT (M)
3130 NEXT K
3135 BIT = 0
3136 IF M < > NN THEN BIT = 1
3142 IF F1 = 0 THEN RETURN
3143 CX = AX + BX
3145 IF F1 = 1 AND BIT = 1 THEN
F2 = 1
3146 IF F2 = 1 THEN RETURN
3147 IF BIT = 1 THEN RETURN
3150 POKE MX, CX
3152 RETURN
3160 REM
3162 REM *****
3164 REM

```

por tanto más la programación científica que a la gráfica, y necesitan algunos conocimientos de matemáticas.

El método visto no utiliza ninguna fórmula, y puede aplicarse a cualquier figura plana. Sin embargo, tiene el defecto del notable tiempo de proceso, debido principalmente a los bucles de control de la página gráfica. Para limitar el tiempo de espera existen dos métodos, que también pueden utilizarse conjuntamente:

1 / utilizar versiones compiladas del programa
2 / circunscribir el área de memoria a explorar a un contorno de la figura.

Por esto, en la subrutina 2000 se piden los límites —según los dos ejes— que deben utilizarse en la exploración de la pantalla; así se evita analizar zonas sin figuras. Una segunda ventaja asociada al uso de estos límites consiste en la posibilidad de analizar más figuras presentes simultáneamente en el vídeo; basta con llamar la

rutina tantas veces como figuras hay, proporcionando para cada una los oportunos límites de exploración. Al introducir este dato debe prestarse mucha atención, puesto que una evaluación errónea podría producir notables errores. También para este tipo de error es posible prevenir una serie de controles. El más sencillo consiste en dibujar en la pantalla un reticulado que proporcione una indicación de las coordenadas. Esta implantación es extremadamente sencilla: se trata de dibujar un rectángulo en los bordes de la pantalla con algunas divisiones de referencia, por ejemplo, cada 10 puntos. La rutina que realiza esta tarea es completamente análoga a la vista para el trazado de los ejes cartesianos o del recuadro para los histogramas. Un segundo método (analítico) consiste en memorizar las coordenadas máximas y mínimas a medida que se crea el dibujo; al final se explora el área rectangular definida entre los

PROGRAMA PARA EL CALCULO DE AREAS

Examinamos algunos ejemplos de aplicación del programa presentado en las páginas 1561 y 1562. La pantalla presenta la situación después de la introducción del valor 3 en respuesta a la línea 1015. El control se transfiere a la rutina 1300, que pide la base, la altura y la posición del triángulo y proporciona el valor analítico del área (1321, 1322).



El programa entra después en la rutina 2000, que calcula el área explorando la memoria vídeo. La 2000 pide los límites de exploración de la pantalla según el eje X y el eje Y. La foto anterior y la de aquí al lado muestran la fase de introducción de estos límites: el eje X se explora desde la posición 50 a la 70; el eje Y desde la posición 70 a la 80. Al final se presentará el valor del área determinado mediante la exploración.



La tercera foto muestra la situación análoga para un cuadrado que tiene lado 50 y está posicionado en 100,70. El área calculada analíticamente vale 2500; el valor determinado por el programa se presentará después de la introducción de los límites de exploración de la pantalla.



La última foto se refiere al caso de un círculo, para el cual el programa se comporta como en los casos anteriores. Es interesante observar cómo en el sistema utilizado también pueden reservarse en el modo gráfico las cuatro líneas inferiores del monitor para el coloquio con el usuario.



valores máximo y mínimo, más un eventual margen. Naturalmente, si existen más figuras en la pantalla, el usuario debe informar al programa cuándo se ha terminado una figura.

La definición de un entorno de exploración limitado para cada figura también protege contra otro tipo de error. En la generación de un dibujo es posible que, a causa de las imprecisiones, el contorno no quede cerrado; en este caso, la exploración y el consiguiente cálculo no terminarían hasta el margen de la pantalla, atribuyendo a aquella faja un área mucho mayor que la real. En cambio, definiendo un contorno máximo, suficientemente próximo al dibujo, se tiene la interrupción del cálculo en cualquier caso.

Sin embargo, la necesidad de tener un contorno cerrado es el motivo del bucle utilizado en la subrutina 1400 para trazar la circunferencia.

El círculo debe trazarse partiendo del ángulo 0 y terminando con el valor angular $2 * \pi = 6.28$ (que equivale a un giro completo), mientras que en el bucle se utilizan los límites -0.05 y 6.3 para garantizar el cierre completo de la figura.

Instrucciones Basic para la memorización de figuras

El método expuesto para la adquisición de una figura no utiliza instrucciones particulares y puede emplearse en todas las máquinas para las cuales se haya indicado las direcciones de la memoria vídeo.

En algunos casos existen instrucciones de alto nivel que ejecutan esta tarea de manera transparente, es decir, no necesitan ninguna indicación que no sea la definición en coordenadas vídeo de la zona a memorizar o a redibujar.

Este tipo de instrucciones está muy ligado a la máquina, después a la sintaxis, y a veces también a las funciones realizadas, que dependen del tipo de ordenador al que se implantan dichas instrucciones. A título de ejemplo, las instrucciones presentes en el ordenador personal Olivetti M20 dedicadas a este objeto son:

POINT detecta el color presente en las coordenadas especificadas
GET memoriza una zona del vídeo
PUT presenta la imagen memorizada con un GET

Hay que tener en cuenta que las instrucciones GET y PUT utilizadas en este empleo no son las inherentes a las funciones I/O: tienen una sintaxis y una finalidad completamente diferentes.

POINT. La sintaxis completa de la instrucción POINT es la siguiente:

A% = POINT(X,Y)

Durante la fase de ejecución, el programa asigna a la variable entera A% un valor numérico que depende del color del punto de coordenadas X,Y. Este valor puede ser 0 o 1 en el caso de un vídeo monocromático (0 = apagado, 1 = iluminado) o comprendido entre 0 y 3 o entre 0 y 7 para un vídeo en colores. Los dos límites (0,3 y 0,7) dependen del tipo de vídeo utilizado (4 u 8 colores).

Los valores de las coordenadas X,Y tanto pueden estar referidos al hardware como ser definidos por el usuario. En este tipo de máquina puede utilizarse un sistema de referencia diferente al que define los puntos de pantalla, y que toma el nombre de **referencia usuario**.

La palabra clave POINT también se utiliza en otra instrucción gráfica con un significado diferente. En esta máquina es posible adoptar dos cursores, uno de texto y otro de gráficos. La gestión del cursor de texto es análoga a la de las otras máquinas, aunque con instrucciones formalmente diferentes, mientras que el cursor de gráficos no es muy difundido. En la forma más sencilla, la instrucción tiene la sintaxis

CURSOR POINT(X,Y)N,M

con N y M que indican respectivamente punto en visión (N = 1) o no en visión (N = 0), y la frecuencia con que se desea que parpadee el cursor (con M = 0 no parpadea).

GET. La sintaxis simplificada es

GET(X₁,Y₁) - (X₂,Y₂),A%(0)

donde X₁,Y₁ y X₂,Y₂ definen los vértices de la parte rectangular del vídeo a memorizar, mientras que A%(0) es el primer elemento de una matriz utilizada por el sistema para memorizar los valores (recuérdese que, en Basic, el índice de una matriz empieza por 0).

Después de la ejecución de la instrucción, en A%(.) hay memorizados todos los bits correspondientes a la sección del vídeo especificada por las coordenadas comprendidas en la instrucción. Además del estado de cada bit simple, A%(.) también contiene las informaciones:

A%(0) base del rectángulo definido por X₁,Y₁ y X₂,Y₂
A%(1) altura del rectángulo
A%(2) flag de vídeo monocromático o de colores

Desde A%(3) en adelante, las posiciones se utilizan para los bits del área vídeo definida. El dimensionado de la matriz A%(.) depende de la extensión en pixels de la zona a memorizar, según la fórmula

Dimensión = ((|base/16| * altura) * K) + 3

donde K vale 1 para monocromático, 2 para 4 colores y 3 para 8 colores, y el coeficiente base/16 debe redondearse siempre por exceso. El ordenador M20 puede dividir toda la pantalla en partes independientes entre sí llamadas ventanas. Cada una de ellas puede utilizarse con la indicación del número de ventana de la que hay que tomar datos. En este caso, la sintaxis completa de la instrucción GET es

GET W%(X₁,Y₁) - (X₂,Y₂), A%(0)

donde W% asume el valor numérico de la ventana (el tema se tratará más adelante).

PUT. La sintaxis completa es:

PUT W%(X₁,Y₁) - (X₂,Y₂),A%(0),V

También en este caso, el parámetro W% (ventana) es opcional. Respecto a la anterior, esta instrucción contiene el nuevo parámetro V, que expresa una operación lógica a realizar entre los números de color contenidos en la matriz y los presentes en el vídeo dentro del rectángulo definido por las coordenadas especificadas.

Si el parámetro V está omitido, la matriz A%(.) se presenta tal como está memorizada, Y viceversa, si V está especificado, la presentación es el resultado de la operación lógica, expresada por V, entre los colores preexistentes y los de A%(.). La opción V puede asumir los valores AND, OR, XOR, NOT, PSET, PRESET. AND, OR, XOR tienen el significado habitual, NOT indica el complemento de los números de color presentes en el vídeo, PSET presenta la matriz tal como está memorizada y PRESET indica el complemento de los números de color de la matriz.

Las ventanas vídeo

En muchas aplicaciones es útil poder dividir la pantalla en zonas gestionadas por separado una de otra. Estas zonas toman el nombre de **ventanas vídeo**, y se comportan como otras tantas pantallas vídeo pequeñas y separadas. El uso más frecuente de las ventanas es el de presentación simultánea de un dibujo de conjunto y de algunos de sus detalles. Por ejemplo, para representar una función puede dividirse la pantalla en dos o tres partes, de las que una, generalmente la mayor, se dedica al gráfico entero, mientras que las otras sirven para analizar en detalle intervalos pequeños.

La utilización de las ventanas vídeo, en los ordenadores en que se han previsto, se realiza en dos fases. En la primera se implantan los parámetros que definen cada ventana (dimensiones, posición, etc.); a continuación puede utilizarse el área de pantalla así definida direccionándola con el número o con otro símbolo asociado a ella en la anterior fase de generación.

Las posibilidades en la gestión de las ventanas vídeo varían mucho según la máquina. Pueden indicarse tres categorías principales:

- máquinas en que las ventanas no están previstas de ninguna manera
- ordenadores que prevén sólo dos ventanas (la primera funciona en modo gráfico y la otra en modo texto)
- ordenadores con una gestión completa de las ventanas, tanto en el número como en el modo (gráfico y texto)

Las dos primeras categorías prácticamente son equivalentes. La presentación de una ventana de texto, a menudo limitada a algunas líneas, sólo puede utilizarse en los casos en que es necesario un coloquio con el usuario, manteniendo la presentación de la página gráfica.

Por ejemplo, éste es el caso de los ordenadores compatibles Apple, que con la instrucción HGR conservan una ventana de texto de cuatro líneas de cabida.

Por tanto, no se trata de una división en ventanas gráficas, aunque puede implantarla el usuario con rutinas relativamente sencillas.

A continuación se describe un método de preparación de estas subrutinas, las cuales pueden aplicarse a la mayoría de ordenadores sin modificaciones, excepto las instrucciones gráficas, que son específicas de cada máquina.

Programa para la generación de ventanas vídeo

Una ventana es una zona de la pantalla, generalmente de forma rectangular o cuadrada, identificada con un símbolo o con un número. Para utilizar esta subdivisión, un programa de gráficos debe ser parametrizado, o sea debe gestionar todos los posicionados y los desplazamientos.

Esta parametrización es necesaria sólo en los casos en que la subdivisión en ventanas se obtiene con programas escritos para el usuario. En las máquinas en las que las ventanas vídeo están previstas a nivel de sistema, la parametrización es transparente al usuario, que sólo debe especificar en qué ventana quiere trabajar.

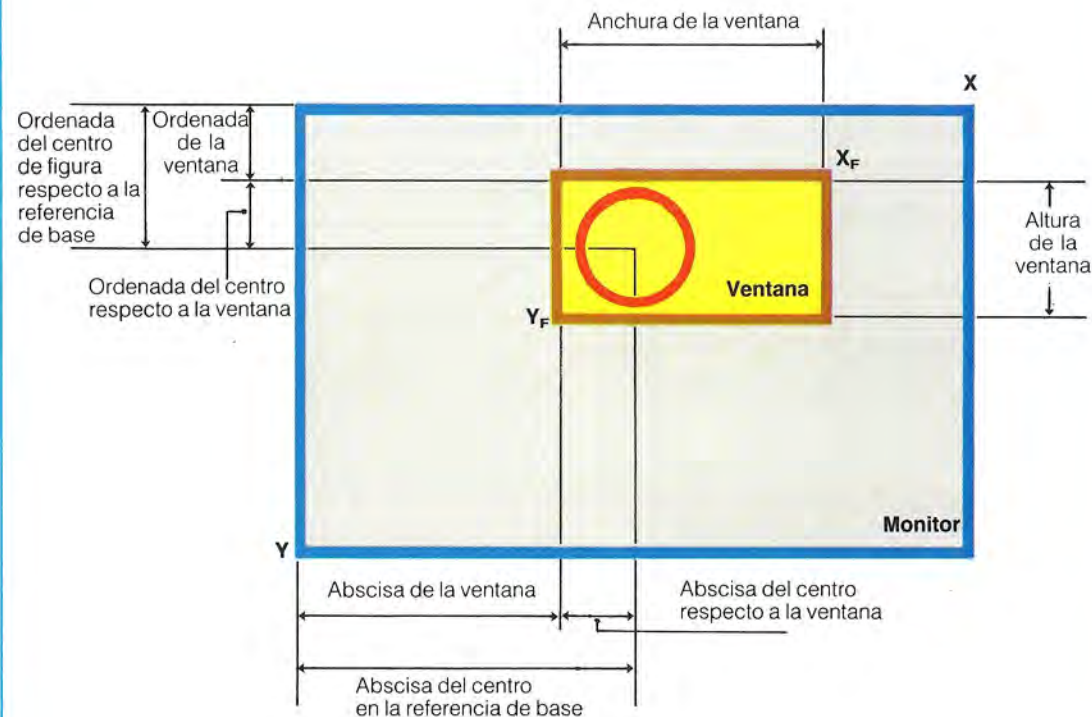
Abajo se ha representado la imagen de una circunferencia referida tanto al sistema de base como a un sistema de referencia coincidente con dos lados de una ventana. En los casos más sencillos, la gestión de un dibujo en el interior de una ventana necesita la construcción de la figura refiriendo las coordenadas al sistema de referencia solidario con la ventana. La presentación de un círculo de radio R se obtiene con un bucle que calcula las coordenadas de cada punto utilizando las fórmulas

$$X = X_0 + R * \text{COS}(A)$$

$$Y = Y_0 + R * \text{SEN}(A)$$

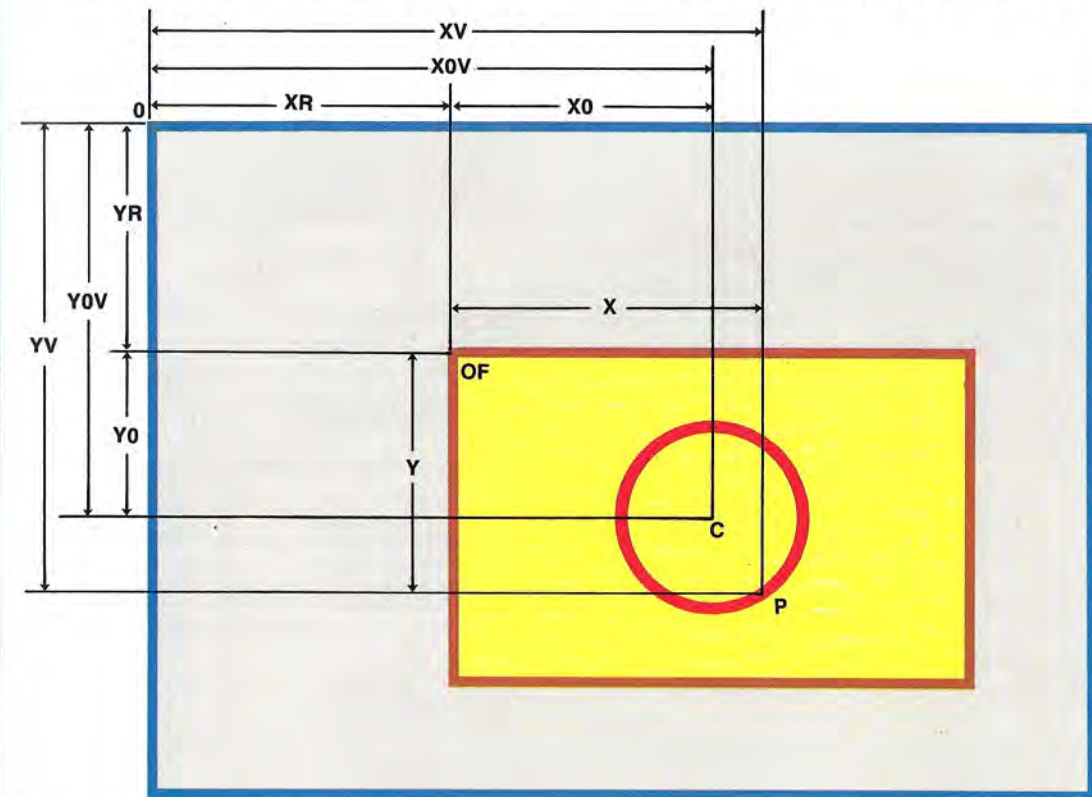
Donde X_0 e Y_0 son las coordenadas del centro.

GESTION DE UNA FIGURA EN UNA VENTANA VIDEO



— Área útil del monitor y referencia de base
 — Ventana y correspondiente sistema de referencia

SIMBOLOGIA UTILIZADA PARA LA GESTION DE LAS VENTANAS VIDEO



Para definir la figura (circunferencia) en la ventana basta con referir las coordenadas del centro a las de la ventana. La simbología utilizada se presenta arriba. Las coordenadas X, Y de un punto general de la circunferencia referida a la ventana deben proporcionarse como coordenadas X_V, Y_V referidas al sistema de base, que es el utilizado por el ordenador para direccionar los desplazamientos del haz electrónico. De la figura se comprueba inmediatamente que

$$X_V = X_R + X$$

$$Y_V = Y_R + Y$$

Análogamente, las coordenadas del centro, si están referidas al sistema de base, pasan a ser:

$$X_{0V} = X_R + X_0$$

$$Y_{0V} = Y_R + Y_0$$

Por tanto, para posicionar el círculo en el interior de la ventana basta sumar a las coordenadas

de cada punto las coordenadas del origen de la referencia solidaria con la ventana.

En este caso particular, las coordenadas de cada punto de la figura están referidas a las de su centro y, por lo tanto, es indiferente realizar la suma para cada punto simple o sólo para el centro. Es decir, el posicionado de una figura en el interior de una ventana se reduce a una sencilla traslación del sistema de referencia. Los dos modos de operar (sobre coordenadas sencillas o sólo sobre las de una referencia) pueden conservarse incluso para figuras que no tienen un centro, siempre que las coordenadas de cada punto estén referidas a un punto fijo, por ejemplo un vértice.

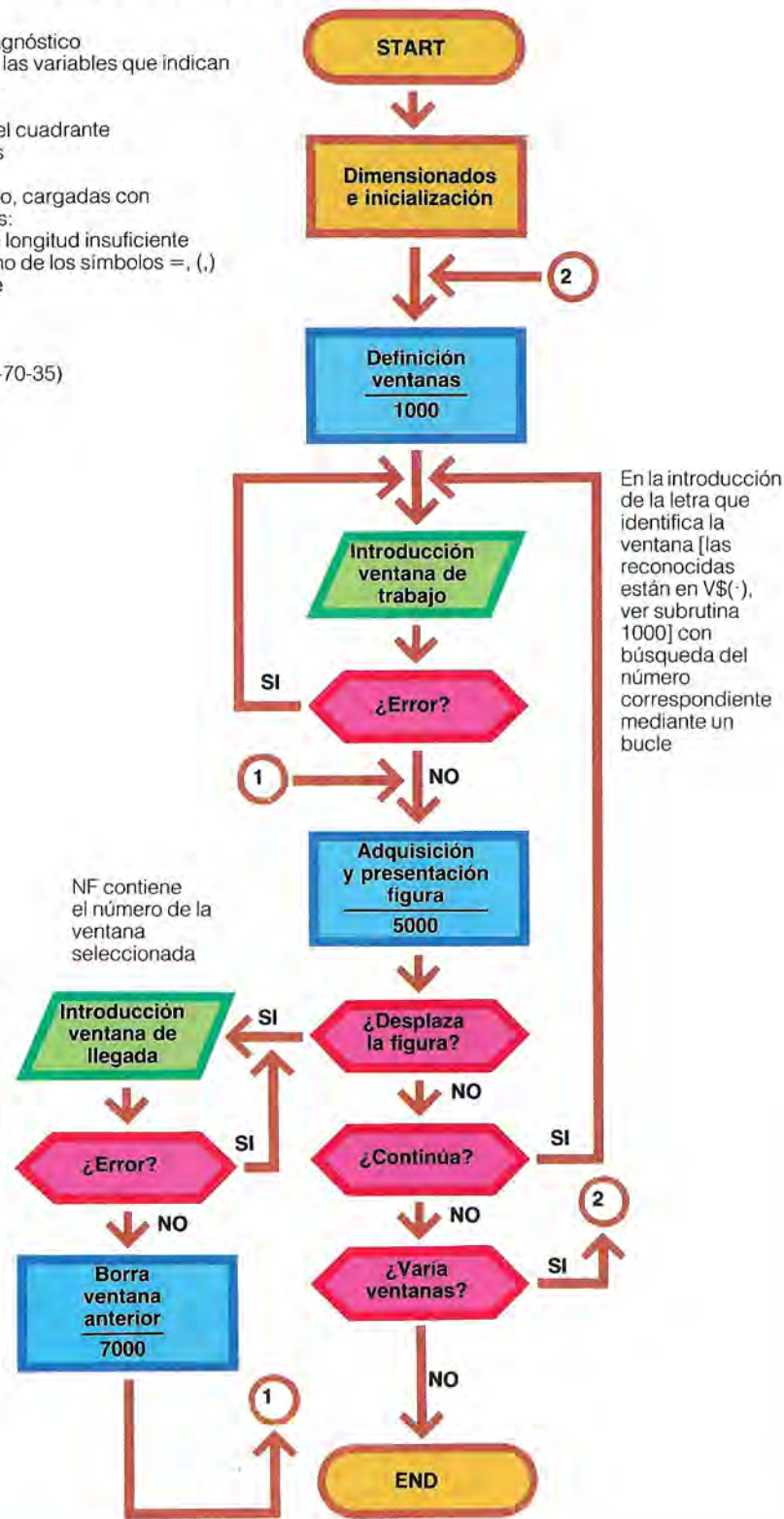
En la figura de la página siguiente se ha representado el diagrama de flujo de un programa de demostración que prevé el uso simultáneo de varias ventanas. En cada ventana es posible dibujar una figura cualquiera referida a un sistema de coordenadas solidarias con la propia ventana. Por tanto, es como si se dispusiera de tantas

GENERACION DE VENTANAS VIDEO

D\$(4) Cadenas diagnóstico
 V\$(16) Nombres de las variables que indican las ventanas
 Q(16) Cuadrante
 P(16) Posición en el cuadrante
 H(16),L(16) Dimensiones

Las cadenas de diagnóstico, cargadas con un DATA, son las siguientes:
 1 / cadena de comando de longitud insuficiente
 2 / error de sintaxis, falta uno de los símbolos =, (,)
 3 / error en la palabra clave
 4 / error en los parámetros

Sintaxis de la instrucción:
 F = WINDOW(1-3-70-35)



pantallas vídeo separadas como cuantas son las ventanas generadas.

El programa está estructurado como un intérprete, puesto que la definición de las ventanas se introduce en forma de cadena; la subrutina 1000 la interpreta, pidiendo los parámetros. La forma elegida coincide con la instrucción Basic más empleada para definir una ventana vídeo en los sistemas que prevén esta implantación. La sintaxis que quiere simular es la siguiente:

- una primera letra, cualquiera, con la que identificar la ventana que se está definiendo
- el símbolo = para informar al sistema que seguirá la palabra clave
- una palabra clave (WINDOW) que define la función deseada (creación de una ventana)
- una serie de parámetros, entre paréntesis, que definen la geometría de la ventana

Los parámetros son:

- q define el cuadrante de vídeo (1, 2, 3, 4)
- p define la posición (de 1 a 4) en el interior del cuadrante
- nnn, mmm valores numéricos, cada uno constituido por tres cifras como máximo, que representan respectivamente la anchura y la altura de la ventana en puntos de pantalla

La cadena a proporcionar, por tanto, tiene la estructura

$$V = WINDOW(1 - 4 - 50 - 20)$$

que se refiere a una ventana identificada por la letra V, que ocupa el primer cuadrante en la posición 4 y tiene las dimensiones de 50 x 20.

Las subrutinas que extraen los parámetros deben estar estructuradas de manera que se descarten eventuales espacios blancos entre las diversas partes de la cadena de comando; por ejemplo, la palabra clave WINDOW puede estar más o menos acercada al símbolo =.

Los valores que definen las dimensiones de la ventana (nnn,mmm) pueden contener indistintamente de 1 a 3 caracteres. Su reconocimiento se realizará basándose en la posición del símbolo - (separación entre campos) y del paréntesis de cierre.

En realidad, para realizar las funciones de gene-

ración de una ventana, la metodología utilizada no es imprescindible. El ejemplo se ha presentado así para mostrar una técnica utilizable en la escritura de un intérprete.

En este caso particular, la palabra clave a reconocer sólo es una, por lo que puede memorizarse como constante con la que realizar la comparación a los fines de diagnóstico. En general puede preverse una matriz que contenga una serie de palabras clave que puedan activar, después de su reconocimiento, otras tantas subrutinas. Básicamente, las modificaciones necesarias son extremadamente sencillas: en la fase de inicialización se cargan en una matriz las palabras clave previstas (de manera análoga a los diagnósticos de error), y en el desarrollo de las 1500 se activa un bucle para reconocer cuál se ha introducido. Finalmente, si el resultado es positivo (palabra reconocida), se prosigue analizando los restantes términos de la cadena, que en este punto se trata como si fuese una instrucción en un lenguaje particular ideado por el usuario. La única dificultad que presenta este método reside en la proliferación de las subrutinas de control y de extracción de los parámetros.

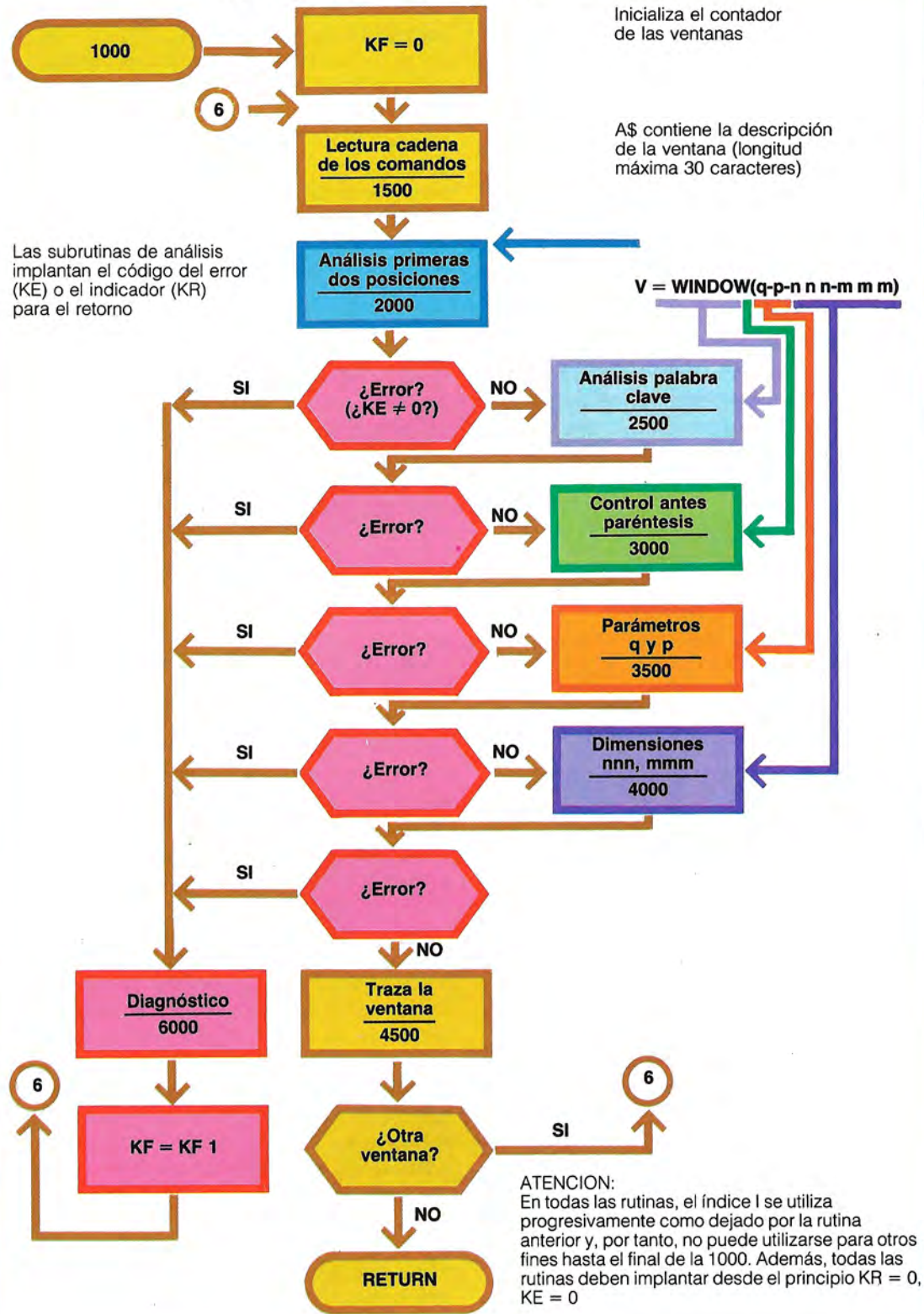
En el caso presentado (una sola palabra clave), los parámetros que siguen no pueden tener otra forma que la prevista, mientras que en un caso general deben llamarse tantas subrutinas de control y de extracción de los parámetros como cuantas sean las posibles alternativas y, por tanto, en número directamente proporcional al de las palabras clave.

Esta estructura, aunque sea de manera muy simplificada, respeta el funcionamiento del intérprete Basic y puede emplearse para construir un lenguaje orientado a los gráficos.

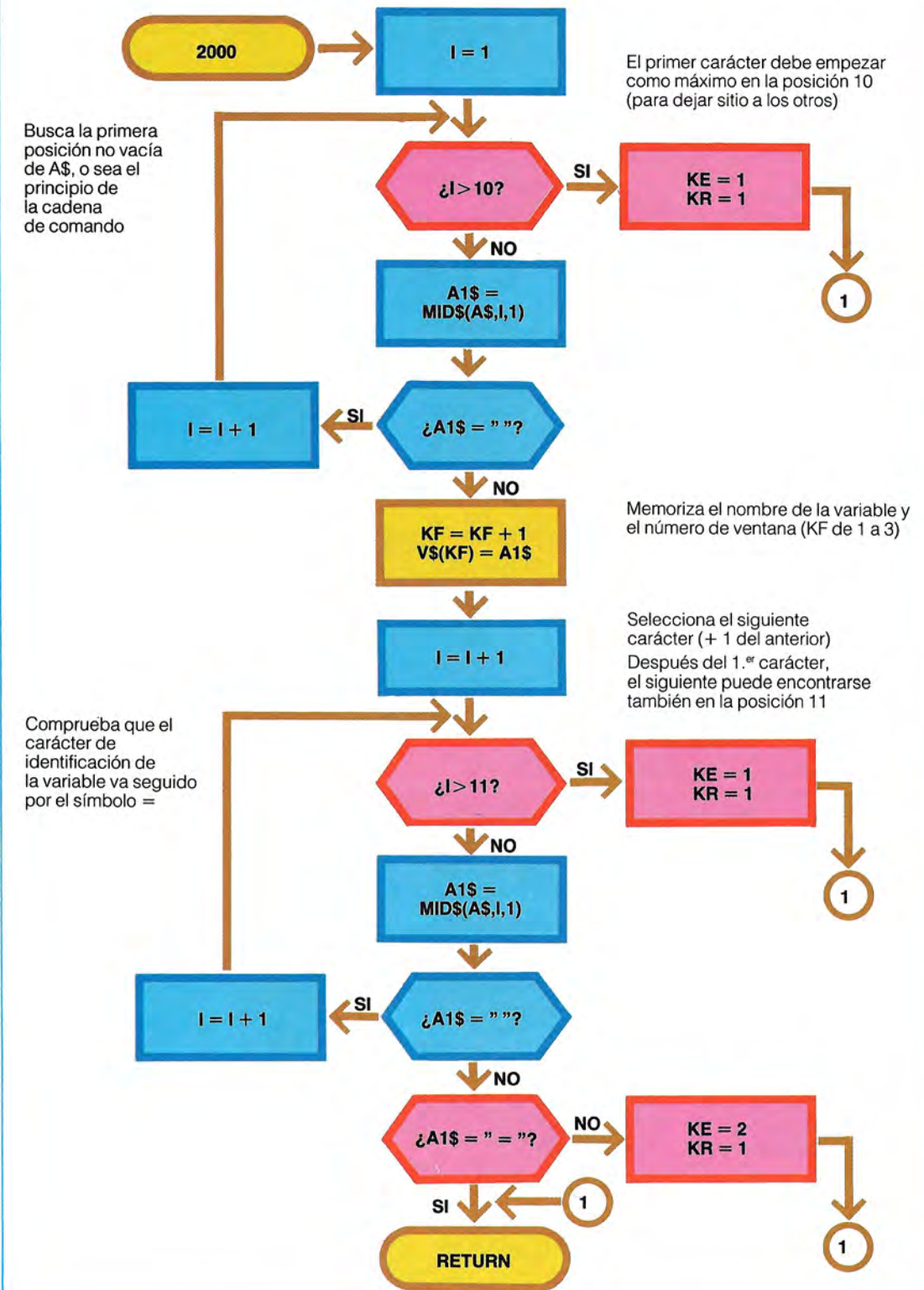
El programa representado sólo tiene la finalidad de demostración, y no contiene controles necesarios que se refieren, por ejemplo, a las dimensiones de la ventana y del dibujo, ni eventuales variaciones de escala. El usuario es el que debe proporcionar valores congruentes durante la introducción de los datos. También en este caso, la modificación es relativamente sencilla, y consiste en comprobar que las coordenadas de cada punto del dibujo caen dentro de la ventana. En caso contrario, la figura queda cortada cerca de los límites de la ventana (esta operación se conoce por «clipping»).

En las páginas que siguen se han representado los diagramas de flujo y el listado del programa.

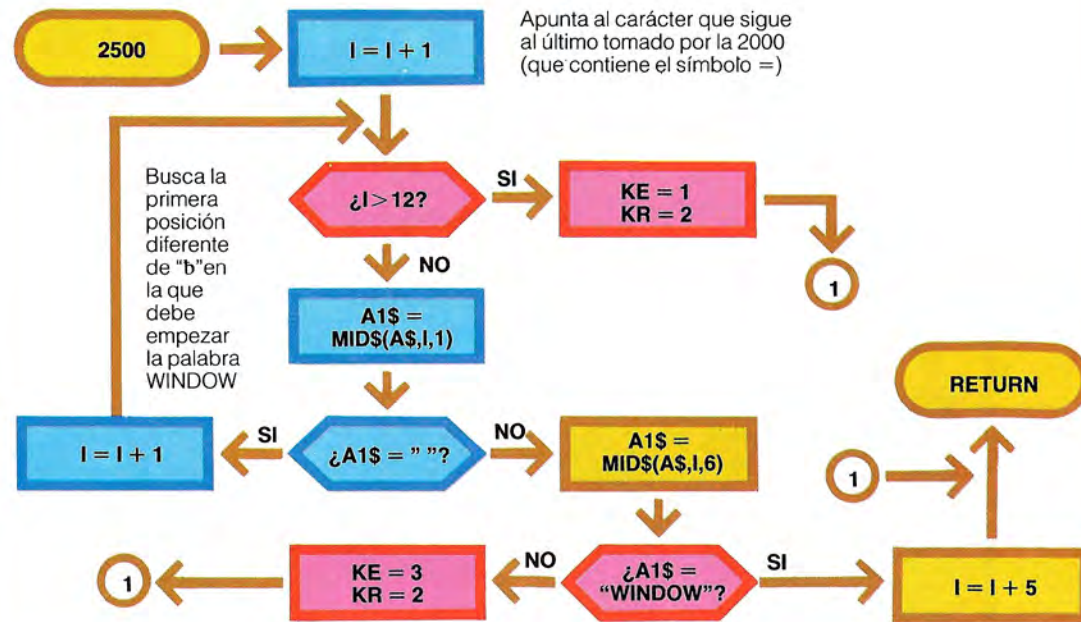
SUBROUTINA DE DEFINICION VENTANAS



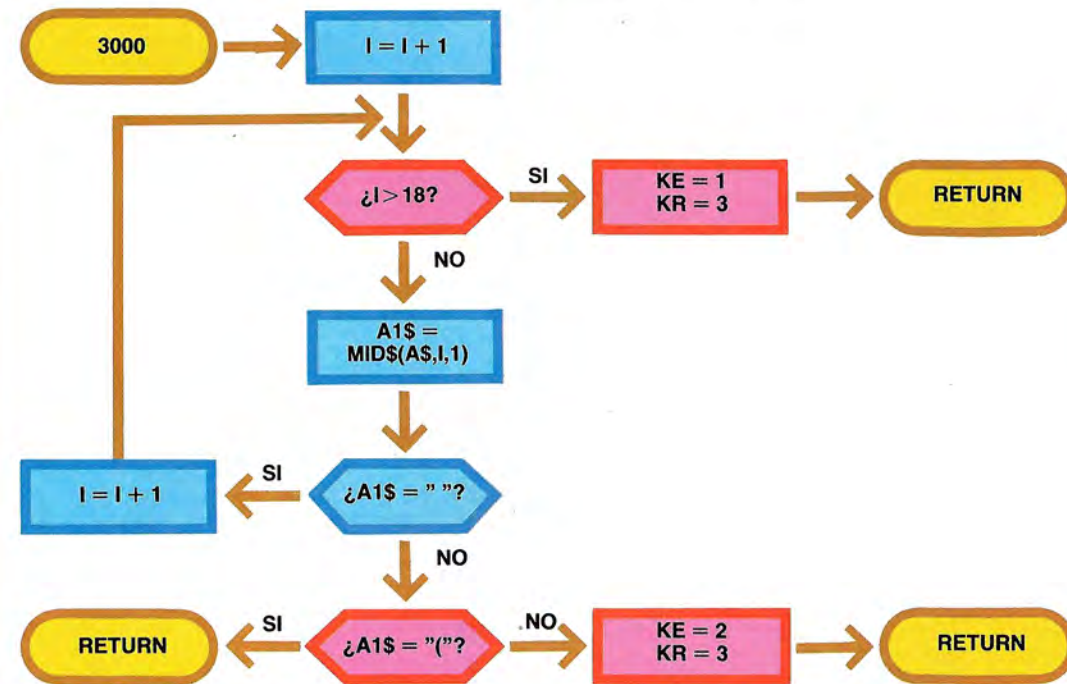
SUBROUTINA DE ANALISIS DE LOS PRIMEROS DOS CARACTERES



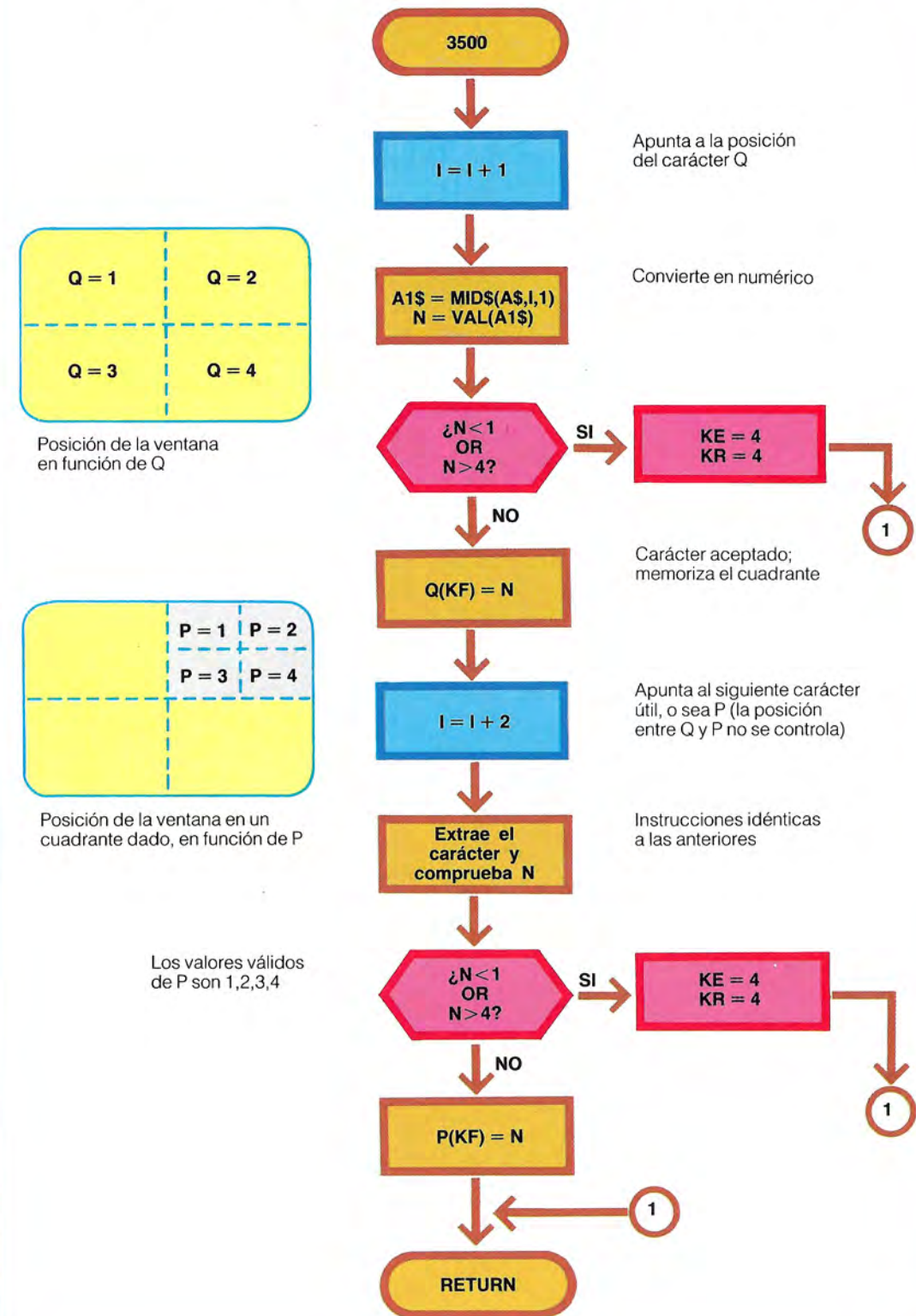
ANALISIS DE LA PALABRA CLAVE



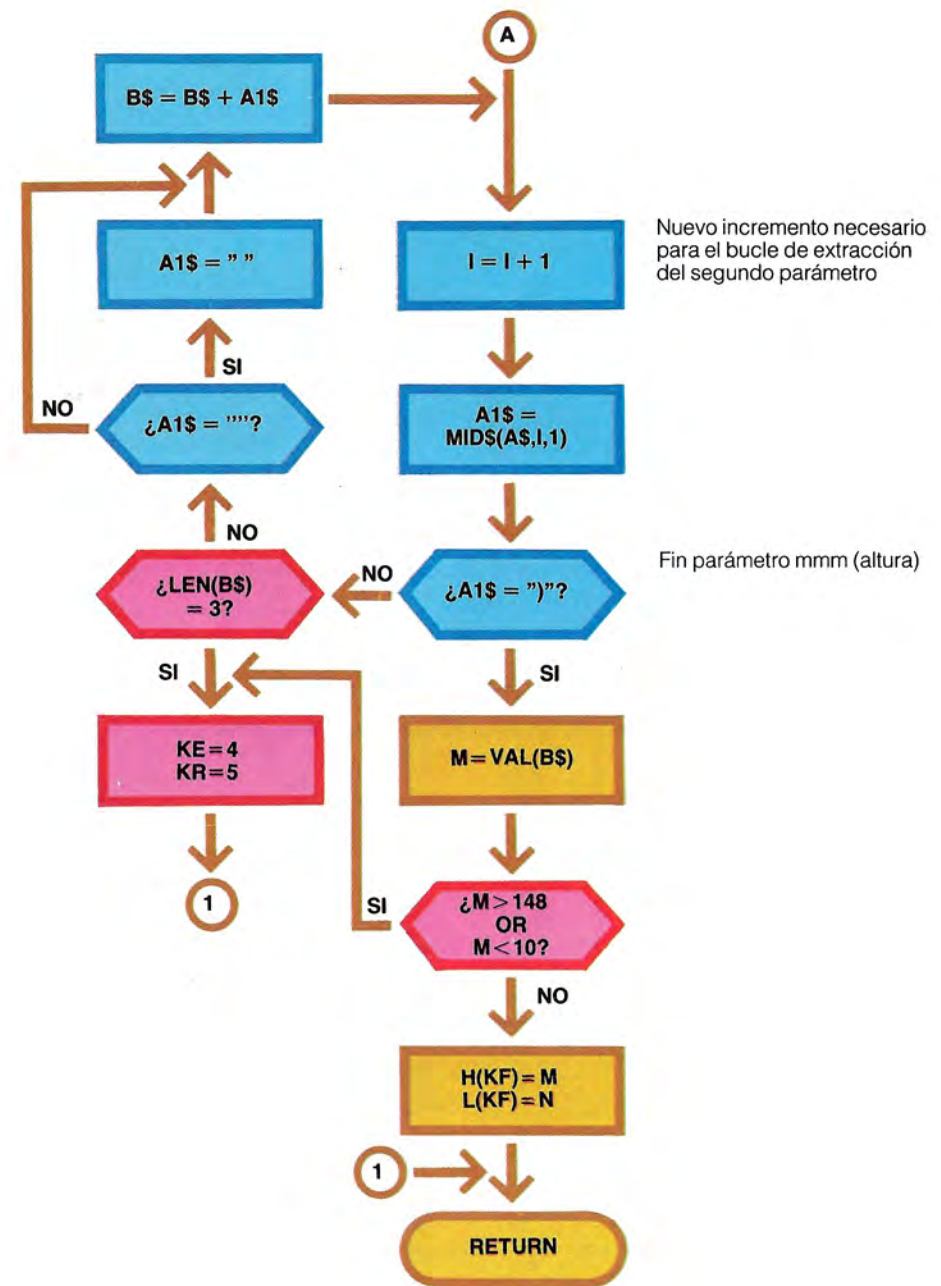
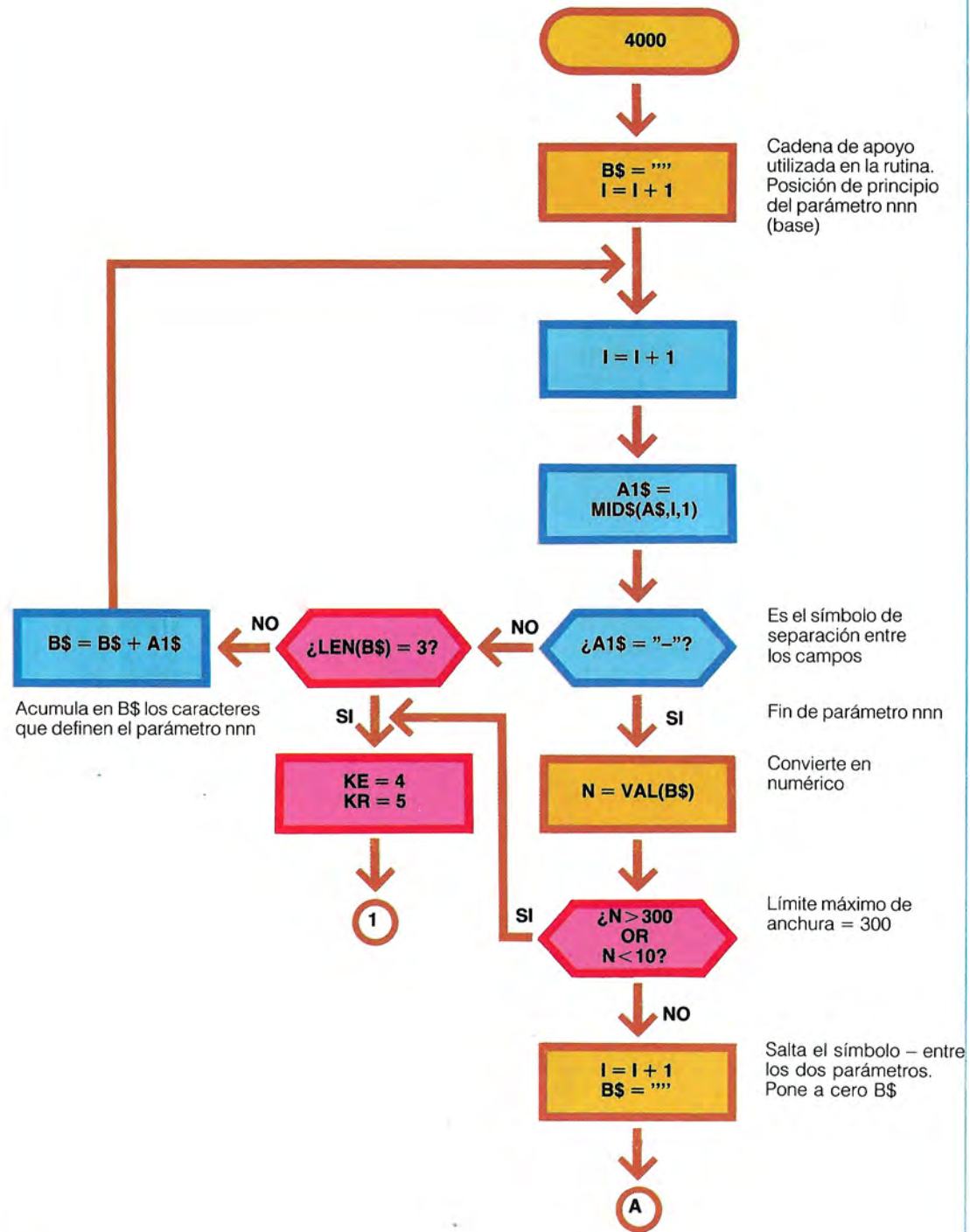
EXISTENCIA PRIMER PARENTESIS



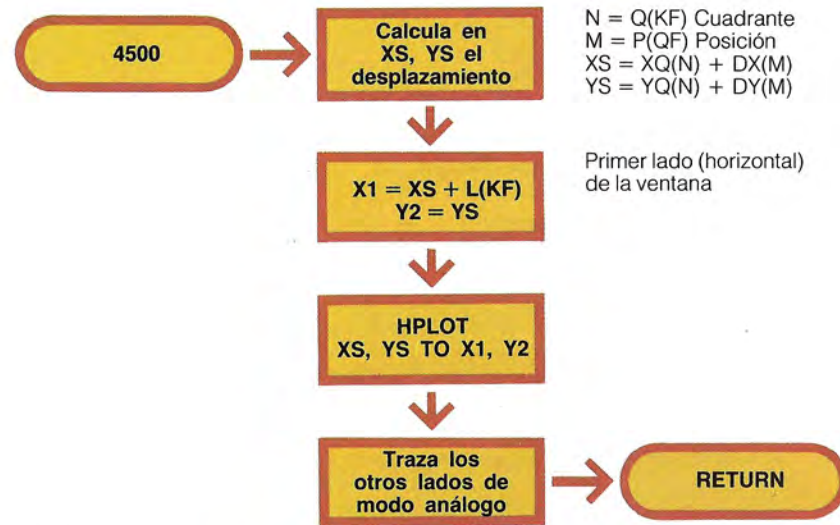
CONTROL PARAMETROS Q Y P (CUADRANTE Y POSICION)



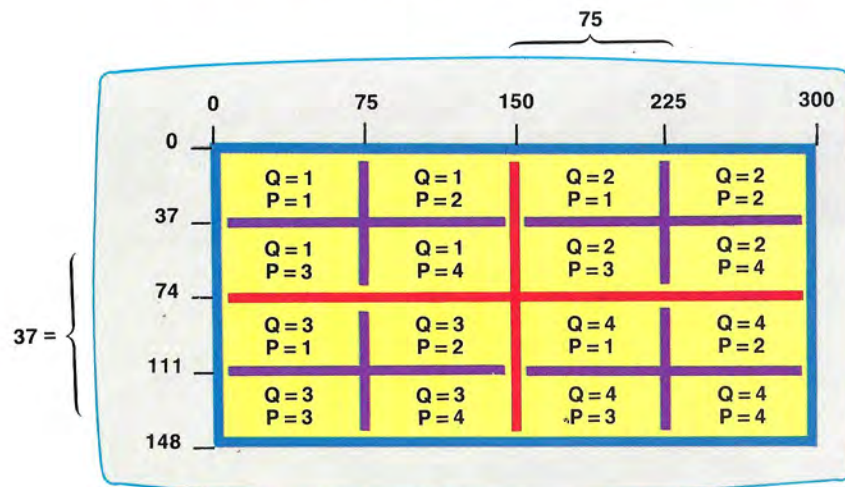
CONTROL DIMENSIONES (PARAMETROS nnn, mmm)



SUBROUTINA DE PRESENTACION DE LA VENTANA



Distribución de la pantalla y coordenadas de los nuevos orígenes en función de los parámetros



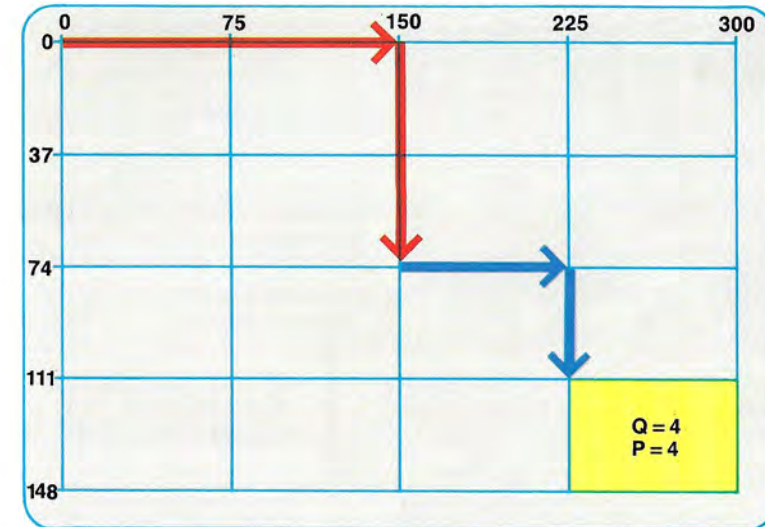
Para cada cuadrante (subdivisiones rojas), la referencia es el extremo superior izquierdo. Así se obtienen los valores que siguen:

Cuadrante	Coordenadas del vértice
1	0,0
2	150,0
3	0,74
4	150,74

que se memorizan (con un DATA) en XQ(4) e YQ(4). Análogamente, el origen de la posición en el interior de un cuadrante está definido por el vértice superior izquierdo [memorizada en DX(4) y en DY(4)] referido al del cuadrante. Los valores son

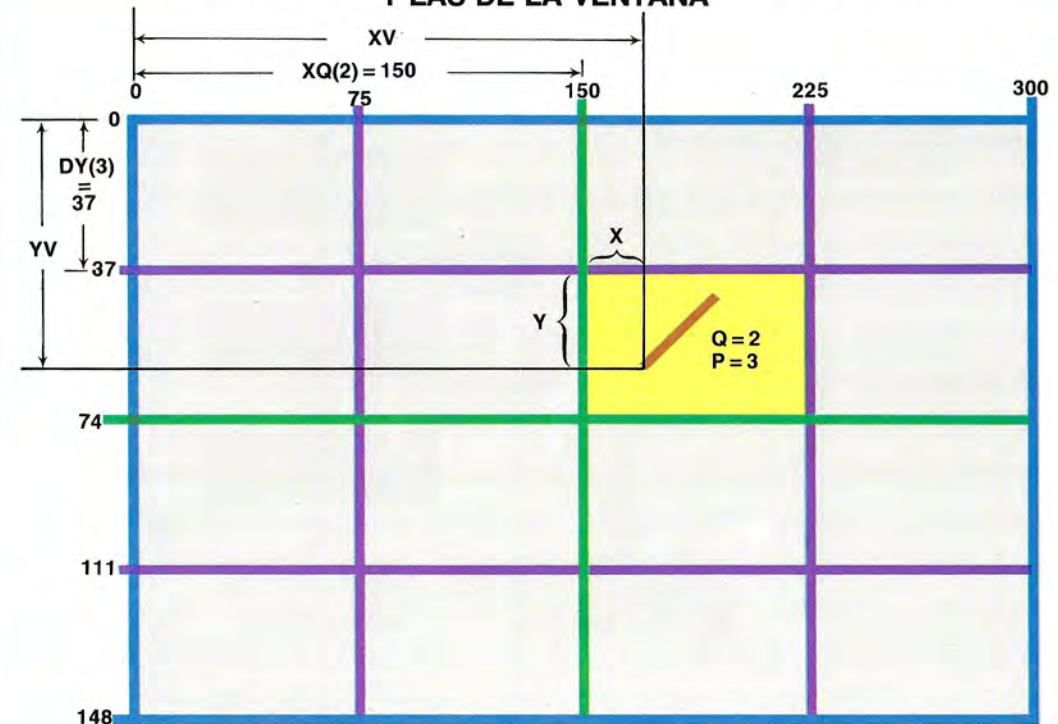
$XQ(\cdot) = 0,150,0,150$
 $YQ(\cdot) = 0,0,74,74$
 $DX(\cdot) = 0,75,0,75$
 $DY(\cdot) = 0,0,37,37$

USO DE LAS COORDENADAS DE CUADRANTE Y DE POSICION



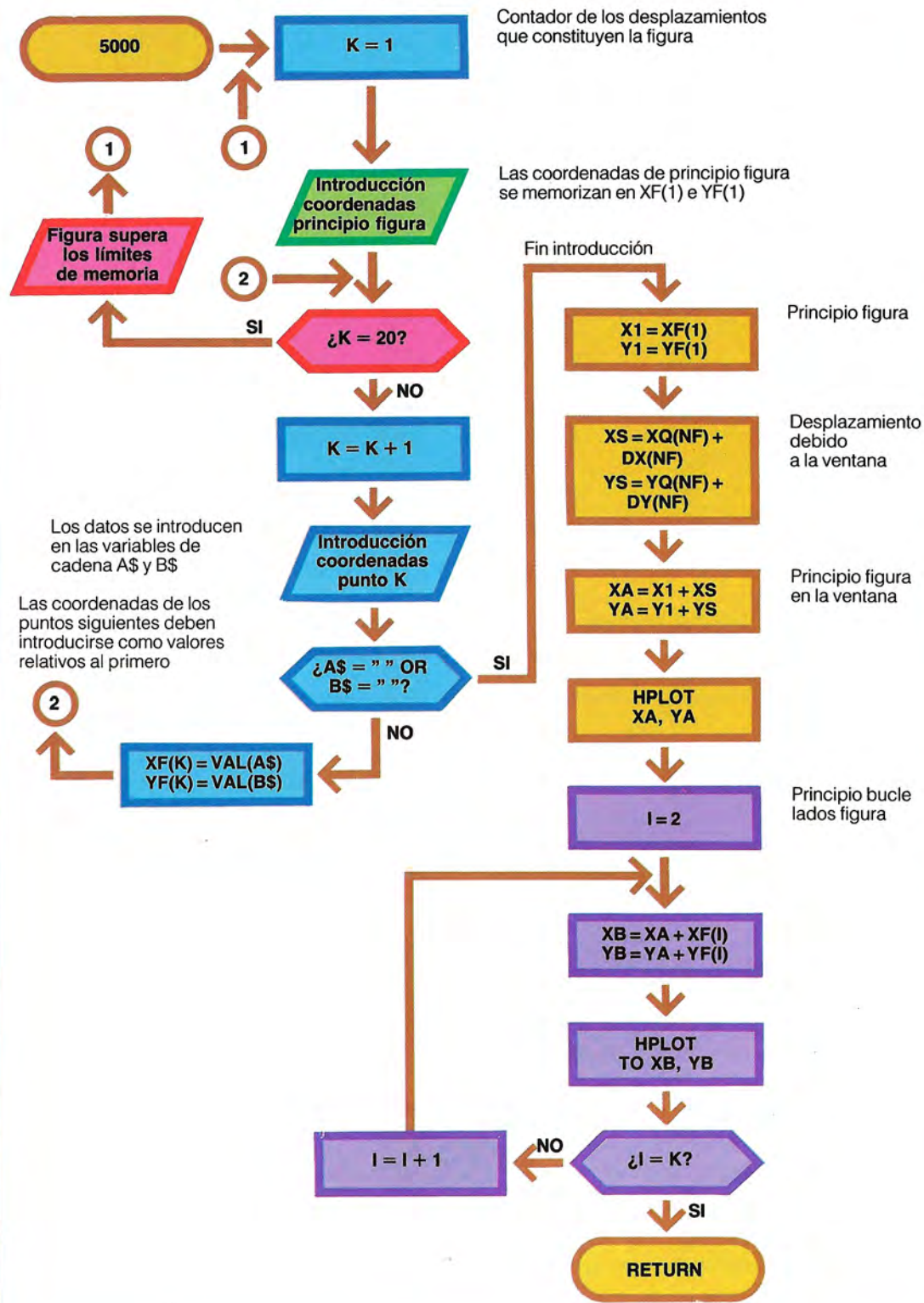
La ventana en el cuadrante 4 en la posición 4 se obtiene desplazándose primero en el cuadrante, o sea a las coordenadas $X = XQ(4)$ e $YQ(4)$ (trazo rojo). Después debe desplazarse, en el interior del cuadrante, a las coordenadas $X1 = DX(4)$ e $Y = DY(4)$ (trazo azul). Así se identifica el extremo superior izquierdo de la ventana. Si se varían las dimensiones de la pantalla, también deben variarse los DATA.

RELACION ENTRE LAS COORDENADAS DE LA FIGURA Y LAS DE LA VENTANA



Las coordenadas reales de un punto (XV,YV) se obtienen sumando las del cuadrante y de la posición a las del punto referidas a la ventana

ADQUISICION Y PRESENTACION FIGURA



CREACION Y GESTION DE LAS VENTANAS VIDEO

```

1  REM ===== 600 HOME
2  REM * PROGRAMA * : VTAB 22
3  REM * DE SIMULACION * : END
4  REM * VENTANAS * 640 GOSUB 120
5  REM * EN PAGINA HGR2 * : GOSUB 7000
6  REM ===== 645 IF Q$ = " " THEN 655
7  : 650 GOSUB 5125
8  : 655 GOTO 540
9  : 1000
10 :
100 HOME 1001 REM =====
100 GOSUB 10000 1002 REM *DEF./VENTANA*
110 GOTO 500 1003 REM =====
115 : 1004
120 REM ===== 1010 KF = 0
130 REM *INT/VENTANA* : HGR2
140 REM ===== 1020 GOSUB 1500
145 : 1030 GOSUB 2000
147 TEXT 1040 IN KE < > 0 THEN 6000
150 HOME 1050 GOSUB 2500
: PRINT "VENTANAS DISPONIBLES !" 1070 GOSUB 3000
160 VTAB 3 1080 IF KE < > 0 THEN 6000
: PRINT I$ 1090 GOSUB 3500
170 FOR J = 1 TO KF 1100 IF KE < > 0 THEN 6000
180 PRINT V$(J); 1110 GOSUB 4000
: HTAB 11 1120 IF KE < > 0 THEN 6000
: PRINT Q(J); 1130 GOSUB 4500
: HTAB 23 1140 TEXT
: PRINT P(J); 1150 HOME
: HTAB 31 : VTAB 10
: PRINT H(J)"*L(J) : INPUT "OTRA VENTANA ? ";RS$
200 NEXT J 1160 IF LEFT$(RS$,1) = "S" THEN 1020
210 VTAB 22 1170 RETURN
: INPUT "INSERTAR EL NOMBRE DE LA VENTANA : ";NM$ 1499
220 FOR J = 1 TO KF 1500 REM =====
230 IF NM$ = V$(J) THEN NF = J 1501 REM *DES./VENTANA*
: J = KF 1502 REM =====
: NEXT J 1503
: GOTO 250
240 NEXT J 1505 TEXT
245 GOTO 210 1510 HOME
250 RETURN : VTAB 10
499 : INPUT "VENTANA : ";A$
: 1520 IF LEN(A$) > 30 THEN 1510
500 REM ===== 1530 RETURN
501 REM *LLAMA/RUTINAS* 2000
502 REM =====
503 : 2001 REM =====
: 2002 REM *CTRL POS. 1-2*
505 TEXT 2003 REM =====
510 GOSUB 1000 2004
520 GOSUB 120
530 GOSUB 5000 2010 KR = 0
540 TEXT : KE = 0
: HOME : I = 1
: VTAB 10 2020 IF I > 10 THEN KE = 1
: INPUT "QUIERE DESPLAZAR LA FIGURA ? ";RS$ : KR = 1
: RETURN
550 IF LEFT$(RS$,1) = "B" THEN 640 2030 A1$ = MID$(A$,I,1)
560 VTAB 10 2040 IF A1$ = " " THEN I = I + 1
: INPUT "QUIERE CAMBIAR DIBUJO ? ";RS$ : GOTO 2020
: RETURN 2050 KF = KF + 1
570 IF LEFT$(RS$,1) = "S" THEN 520 : V$(KF) = A1$
580 VTAB 10 2060 I = I + 1
: INPUT "QUIERE VARIAR LAS VENTANAS ? ";RS$ 2070 IF I > 11 THEN KE = 1
: RETURN 2070 IF I > 11 THEN KE = 1
590 IF LEFT$(RS$,1) = "S" THEN 500 : RETURN
  
```

```

2080 A1# = MID# (A#,I,1)          4000
2090 IF A1# = " " THEN I = I + 1
: GOTO 2070
2100 IF A1# = "-" THEN RETURN
2110 KE = 2
: KR = 1
: RETURN
2499
:
2500 REM *****
2501 REM *CTRL WINDOW*
2502 REM *****
2503
:
2510 KR = 0
: KE = 0
: I = I + 1
2520 IF I > 12 THEN KE = 1
: KR = 2
: RETURN
2530 A1# = MID# (A#,I,1)
2540 IF A1# = " " THEN I = I + 1
: GOTO 2520
2550 A1# = MID# (A#,I,6)
2560 IF A1# = "WINDOW" THEN I = I + 5
: RETURN
2570 KE = 3
: KR = 2
: RETURN
3000
:
3001 REM *****
3002 REM * CTRL "(" *
3003 REM *****
3004
:
3010 KE = 0
: KR = 0
: I = I + 1
3020 IF I > 18 THEN KE = 1
: KR = 3
: RETURN
3030 A1# = MID# (A#,I,1)
3040 IF A1# = " " THEN I = I + 1
: GOTO 3020
3050 IF A1# = "(" THEN RETURN
3060 KE = 2
: KR = 3
: RETURN
3500
:
3501 REM *****
3502 REM * CTRL Q/P *
3503 REM *****
3504
:
3510 KR = 0
: KE = 0
: I = I + 1
3520 A1# = MID# (A#,I,1)
3530 N = VAL (A1#)
3540 IF N < 1 OR N > 4 THEN KE = 4
: KR = 4
: RETURN
3550 Q(KF) = N
: I = I + 2
3560 A1# = MID# (A#,I,1)
: N = VAL (A1#)
3570 IF N < 1 OR N > 4 THEN KE = 4
: KR = 4
: RETURN
3580 P(KF) = 4
: RETURN

```

```

4001 REM *****
4002 REM *CTRL PARAMET.*
4003 REM *****
4004
:
4010 KE = 0
: KR = 0
: B# = ""
: I = I + 1
4020 I = I + 1
: A1# = MID# (A#,I,1)
4030 IF A1# = "-" THEN 4070
4040 IF LEN (B#) = 3 THEN KE = 4
: KR = 5
: RETURN
4050 B# = B# + A1#
: GOTO 4020
4070 N = VAL (B#)
: IF N > 300 OR N < 10 THEN KE = 4
: KR = 5
: RETURN
4080 B# = ""
4090 I = I + 1
: A1# = MID# (A#,I,1)
4100 IF A1# = ")" THEN 4130
4110 IF LEN (B#) = 3 THEN KE = 4
: KR = 5
: RETURN
4115 IF A1# = "" THEN A1# = " "
4120 B# = B# + A1#
: GOTO 4090
4130 M = VAL (B#)
: IF M > 148 OR M < 10 THEN KE = 4
: KR = 5
: RETURN
4140 H(KF) = M
: L(KF) = N
: RETURN
4500
:
4501 REM *****
4502 REM *DIB. VENTANA*
4503 REM *****
4504
:
4505 POKE - 16304,0
: POKE - 16297,0
: POKE - 16299,0
4507 HCOLOR= 3
4510 N = Q(KF)
: M = P(KF)
4520 XS = XQ(N) + DX(M)
: YS = YQ(N) + DY(M)
4530 X1 = XS + L(KF)
: Y2 = YS
4535 IF X1 > 279 THEN X1 = 279
4537 IF Y2 > 191 THEN Y2 = 191
4540 H$PLOT XS,YS TO X1,Y2
4550 Y2 = YS + H(KF)
4555 IF Y2 > 191 THEN Y2 = 191
4560 H$PLOT TO X1,Y2
4570 X1 = XS
4575 IF X1 > 279 THEN X1 = 279
4577 H$PLOT TO X1,Y2
4580 Y2 = YS
4585 IF Y2 > 191 THEN Y2 = 191
4587 H$PLOT TO X1,Y2
4600 GET Q#
: RETURN
5000
:

```

```

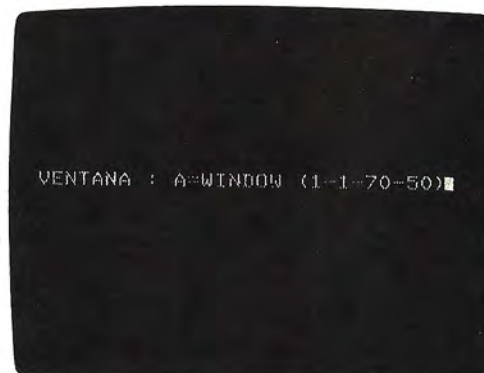
5001 REM *****
5002 REM *REPR. FIGURA*
5003 REM *****
5004
:
5005 TEXT
5010 K = 1
5020 HOME
: VTAB 10
: PRINT "INSERTAR COORDENADAS INI
CIALES"
5030 VTAB 10
: HTAB 30
: PRINT "X / Y "
5040 VTAB 11
: HTAB 31
: INPUT "" ; A#
5050 VTAB 11
: HTAB 35
: INPUT "" ; B#
5052 IF A# = "" OR B# = "" THEN 5020
5055 GOTO 5110
5060 IF K = 20 THEN 5700
5070 K = K + 1
5080 VTAB 10
: HTAB 21
: PRINT "PUNTO " K
5090 VTAB 11
: HTAB 31
: INPUT "" ; A#
5100 VTAB 11
: HTAB 35
: INPUT "" ; B#
5110 IF A# = "" OR B# = "" THEN K = K
- 1
: GOTO 5126
5120 XF(K) = VAL (A#)
: YF(K) = VAL (B#)
: IF XF(K) > 278 OR XF(K) < 1
THEN 5000
5121 IF YF(K) > 191 OR YF(K) < 1
THEN 5000
5122 GOTO 5060
5123
:
5124 REM >> DIBUJA <<
5125
:
5126 HCOLOR = 3
5127 POKE - 16304,0
: POKE - 16297,0
: POKE - 16299,0
5130 X1 = XF(1)
: Y1 = YF(1)
5140 XS = XQ(Q(NF)) + DX(P(NF))
: YS = YQ(Q(NF)) + DY(P(NF))
5145 CF = NF
5150 XA = X1 + XS
: YA = Y1 + YS
: I = 2
5155 H$PLOT XA,YA
5160 XB = XF(I) + XS
: YB = YF(I) + YS
5170 REM
5175 IF XB > = XS + L(NF) THEN XB = XS
+ L(NF) - 1
5176 IF YB > = YS + H(NF) THEN YB = YS
+ H(NF) - 1
5180 H$PLOT TO XB,YB
5190 IF I = K THEN GET Q#
: RETURN
5200 I = I + 1
: GOTO 5160
5700 TEXT
: HOME
: VTAB 10
5710 PRINT "MAX 10 PUNTOS ! " :
: GET Q#
5720 GOTO 5010
6000
:
6001 REM *****
6002 REM *DIAGNOSTICO*
6003 REM *****
6004
:
6010 PRINT CHR# (7)
6020 VTAB 20
: PRINT D#(KE);
: GET Q#
6030 VTAB 20
: PRINT BK#
6040 KF = KF - 1
: GOTO 1020
6999
:
7000 REM *****
7001 REM *BORRA FIG*
7002 REM *****
7003
:
7010 MF = NF
: NF = CF
7020 HCOLOR= 0
: GOSUB 5127
7030 NF = MF
: RETURN
10000
:
10001 REM *****
10002 REM *INICIALIZA*
10003 REM *****
10004
:
10005 HIMEM: 16383
: HGR2
: TEXT
10010 DIM D#(4),V#(16),U(16),P(16),H(16)
,L(16),XF(20),YF(20)
10020 I# = "NOMBRE / CUADRANTE / POSICI
ON / DIMENS."
10025 BK# = ""
: FOR J = 1 TO 30
: BK# = BK# + " "
: NEXT J
10030 FOR J = 1 TO 4
: READ XQ(J),YQ(J),IX(J),DY(J)
: NEXT J
10040 FOR J = 1 TO 4
: READ D#(J)
: NEXT J
10050 RETURN
10100 DATA 0,0,0,0,150,0,75,0,0,74,0,3
7,150,74,75,37
10110 DATA "DATOS INSUFICIENTES", "ERROR
DE SINTAXIS", "PALABRA CLAVE ERRO
NEA","ERROR EN LOS PARAMETROS"

```

GENERACION Y GESTION DE VENTANAS VIDEO

En esta página presentamos algunos ejemplos de funcionamiento del programa representado en las páginas 1579 a 1781.

Al lado puede verse la fase de definición de una ventana de nombre A, altura 50 puntos de pantalla, anchura 70 puntos de pantalla, posicionada en 1,1 (arriba a la izquierda). La palabra clave va seguida de dichos parámetros en orden inverso.

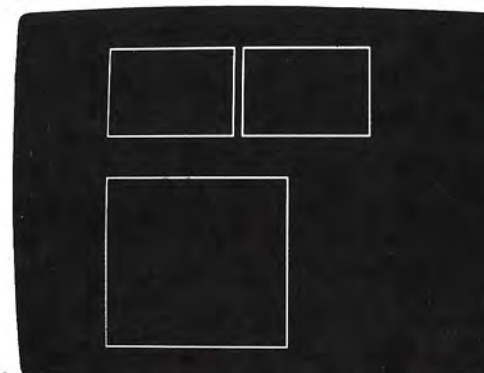


Después de la definición de otras dos ventanas, el programa ha presentado el directorio de las ventanas.

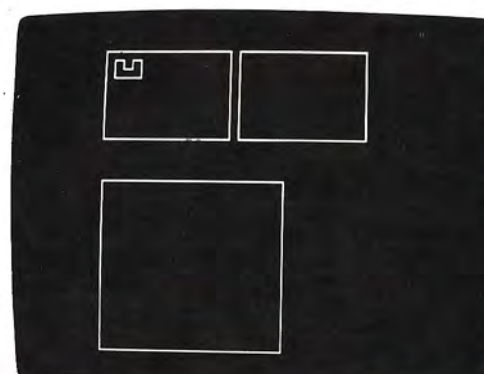
Insertando el nombre de una ventana cualquiera de las definidas se obtiene su activación.



El usuario ha elegido la activación de las tres ventanas definidas anteriormente, que se presentan en la pantalla.



En la última foto se ha insertado una figura gráfica en la ventana A. Utilizando los comandos de desplazamiento, la misma figura puede posicionarse en una cualquiera de las tres ventanas activadas.



El ordenador y la vela

Todo el mundo conoce o por lo menos ha oído hablar de la famosa Copa América, el codiciado trofeo conquistado el 22 de agosto de 1851 por la goleta norteamericana America durante una regata realizada cerca de la isla de Wight.

Fue la primera victoria de la vela del Nuevo Continente sobre la del Viejo.

La copa, que en aquel tiempo se llamaba de las Cien Guineas, fue rebautizada Copa America, y desde entonces, por 132 años consecutivos y a pesar de épicos desafíos que han ido aumentando su fama, ha permanecido en manos de los norteamericanos.

La última edición de la Copa América (1983), convertida ahora en un símbolo que va mucho más allá de un trofeo, representó un giro histórico en la vida de esta competición desde muchos puntos de vista, ante todo por la victoria de Australia con su Liberty, que ha roto la imbatibilidad norteamericana; en segundo lugar, por el creciente interés manifestado también por los no adeptos a este deporte.

Otro factor que ha contribuido no poco a cam-

biar el cariz de la regata ha sido el creciente empleo de la electrónica en general, y en particular del ordenador.

El área de aplicación de los calculadores se ha ampliado desde las primeras utilizaciones off-line como simples instrumentos de cálculo, de soporte del diseñador naval y del deporte de la vela, hasta la utilización durante las pruebas en el mar y en las regatas para la adquisición de los datos de a bordo y para su sucesivo proceso. Por tanto, se está produciendo una verdadera explosión informática a bordo de estos veleeros de carreras cada vez más sofisticados.

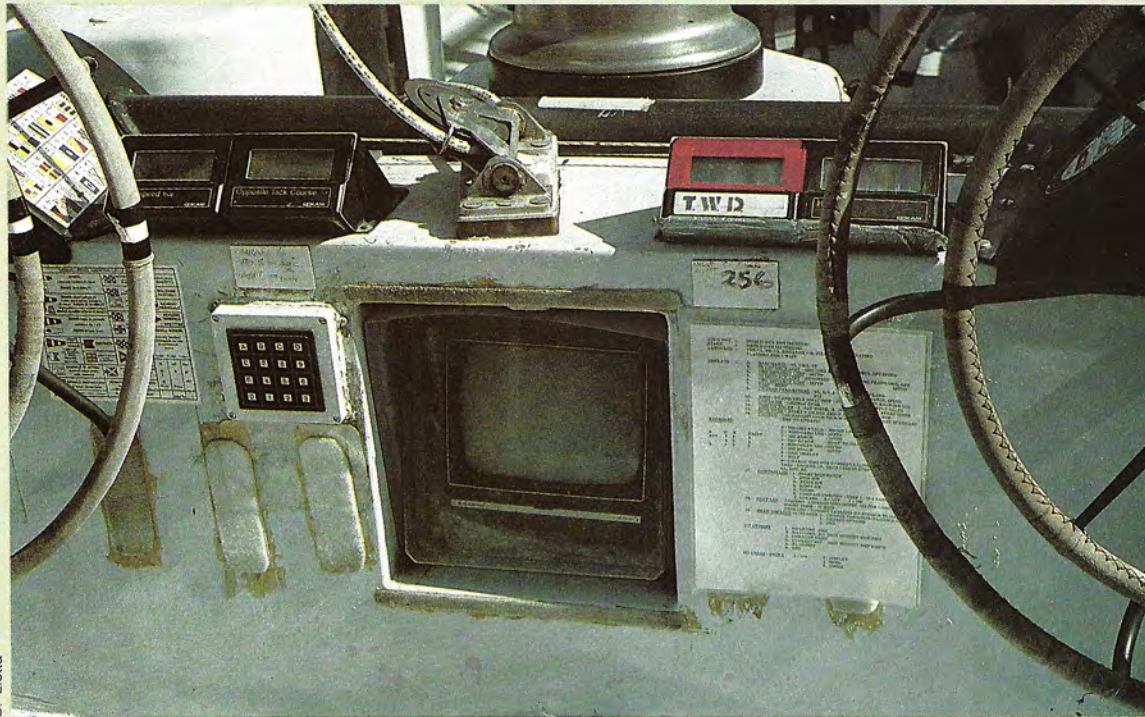
El computador hizo su primera aparición en escena en la Copa América en 1964 con la introducción de un aparato que puede calcular el ángulo real del viento. Los instrumentos de a bordo que miden la dirección y la velocidad del viento presentan de hecho datos sobre el viento aparente, o sea del viento real combinado con el viento de velocidad creado por el propio barco; los datos referentes al viento real deben obtenerse mediante cálculos vectoriales.

A partir de este momento se pasó del empleo de los calculadores como simples instrumentos

Una imagen de la salida de la Copa América.



B. Liotta



B. Liotta

La cubierta del Victory. En primer plano, entre los timones, pueden verse el monitor del ordenador de a bordo, el teclado de mando y los display de cristales líquidos para la presentación de los datos.

de cálculo, dedicados a la solución de problemas de navegación, a su empleo en el centro de un complejo sistema de adquisición, proceso y control de los datos fundamentales de la nave para conseguir la determinación de los correspondientes parámetros de las prestaciones. El enorme desarrollo tecnológico de estos últimos años en el campo de la electrónica, en particular en las dimensiones, las capacidades, la velocidad de cálculo y los costos, ha contribuido mucho a acelerar el proceso de integración de los calculadores como instrumento fundamental a bordo de los 12 metros.

La edición de la Copa América de 1980, en Newport, señaló el principio de una nueva estrategia en el uso del procesador. La novedad ya no consistió en la presencia de un ordenador a bordo, sino en el soporte externo de los potentes sistemas de cálculo instalados en tierra.

Durante los entrenamientos y la regata, todos los datos de interés del velero procedentes de los sensores de a bordo (por ejemplo, velocidad y dirección de la nave, velocidad y dirección del viento, etc.), además de ser presentados en los cuadrantes de los instrumentos de a bordo para su utilización inmediata por la tripulación, eran

adquiridos y procesados por el calculador de a bordo, que los memorizaba continuamente en disco o en cinta magnética. Al final de la jornada de entrenamiento o al final de la regata, los datos se llevaban a tierra y se introducían en el calculador del centro, para poder analizarlos con el fin de obtener el mayor número posible de informaciones respecto a las prestaciones del barco.

En la misma edición de 1980 se asistió a otra novedad importante: la introducción de la telemática para transmitir directamente los datos desde la nave al centro de proceso en tierra. El Independence Syndicate (syndicate es el equivalente norteamericano de nuestro término consorcio) del New York Yacht Club, propietario del defender Clipper (defender significa defensor, y es el nombre general con que se llaman —o se llamaban— los 12 metros norteamericanos, puesto que debían defender la copa que habían conquistado), había adoptado este nuevo método para hacer más rápido y aprovechable el proceso de datos. De esta manera no había que esperar la vuelta a la base de la embarcación para poder disponer de los datos obtenidos; como estos últimos eran continuamente

transmitidos vía radio directamente por el calculador de a bordo al de tierra, su proceso se producía al mismo paso que los acontecimientos, y al enviar al velero los procesos y el análisis del calculador, ya quedaban disponibles para un examen inmediato por la tripulación. Esta innovación pronto se reveló de una gran ayuda, especialmente en la fase de prueba de las nuevas embarcaciones. La tripulación, que debe aprender a conocer un barco en lo referente a sus prestaciones en el mar, puede encontrar en el calculador una ayuda inestimable y un medio de acelerar al máximo la fase de aprendizaje. El reto de 1983 confirmó la tendencia del uso cada vez más difundido del calculador de a bordo y de los sistemas de proceso de soporte en tierra. Sin embargo, hay que decir que uno de los sistemas más avanzados vistos en esta edición estaba instalado a bordo de la embarcación inglesa Victory 83. Éste es precisamente el sistema que utilizaremos como ejemplo para ilustrar con un poco más de detalle la utilización del ordenador a bordo de un 12 metros. En términos generales, el sistema de a bordo puede subdividirse en varios componentes.

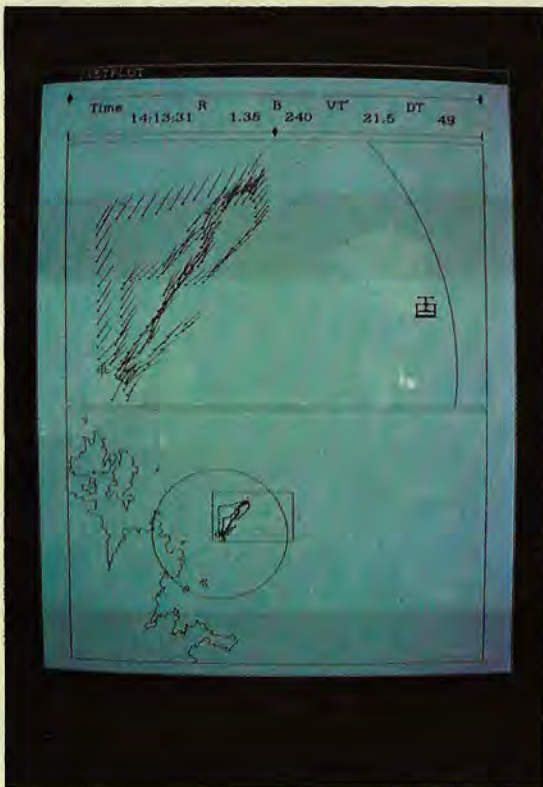
El sistema de detección, tratamiento y presentación de datos está compuesto por los transductores y los sensores que miden las magnitudes de importancia fundamental para el gobierno del velero (por ejemplo, la velocidad y la dirección del viento, la velocidad y la ruta del velero, el ángulo del timón, la posición, etc.), por las unidades de interfaz y por un microprocesador, el cual tiene la misión de efectuar el acondicionamiento de los datos en bruto y algunos procesos sencillos. Todos los datos así obtenidos se visualizan en oportunos presentadores de cristales líquidos para la utilización inmediata por parte de la tripulación y, a través de un interfaz, se envían al procesador de a bordo. El sistema de adquisición y de proceso de datos está constituido por el calculador de a bordo que, mediante el software de sistema, puede realizar procesos particulares, correlacionados de datos, producción de nuevos resultados derivados de los datos de los sensores y presentación y cotejo de datos y resultados.

Hasta ahora, por una serie de motivos más históricos que técnicos, la capacidad del sistema de proceso de a bordo se ha mantenido limita-

El sistema principal de adquisición, proceso y memorización de datos instalado en la barca de apoyo del Victory.



B. Liotta



B. Liotta

Presentación del campo de regata en el vídeo gráfico y reconstrucción de la ruta del Victory. En la parte superior, el campo ampliado.

da, y se ha preferido confiar las funciones principales de proceso, análisis estadístico, producción de funciones, memorización, etc., a un ulterior sistema de cálculo, más potente que el de a bordo e instalado en tierra o, como en el caso del Victory, en una barca de apoyo (una embarcación a motor con cabina, cuya misión principal es la de asistir constantemente al 12 metros hasta pocos minutos antes del principio de la regata y después del final de ésta). El recorrido de los datos no se detiene pues a bordo de la embarcación, sino que prosigue en transmisión de radio hasta el sistema de cálculo principal, y la transmisión automática de los datos vía radio ya es una norma universalmente aceptada en el mundo de los 12 metros.

El sistema principal de adquisición, proceso y memorización de datos está compuesto por el computador, la unidad de memoria de masa, el vídeo de gráficos, la impresora gráfica y el teclado. Éste es el núcleo central de todo el sistema, en el que se realizan los principales análisis de los datos, donde reside el banco de datos de

la embarcación y donde, gracias a los instrumentos de que se dispone, es posible controlar de la manera más ventajosa las prestaciones del velero.

Los datos procedentes de los sensores de a bordo son recibidos con regularidad y procesados en tiempo real cuando el computador principal está predispuesto para esta actividad. De esta manera, el operador tiene la posibilidad de tener bajo control toda la situación, de señalar vía radio a la tripulación eventuales anomalías que puedan producirse (sólo en las pruebas, puesto que durante la regata no está permitido comunicar con los 12 metros) y memorizar, si se considera oportuno, todos los datos adquiridos. La próxima edición de la Copa América tendrá lugar en Perth, Australia, en 1987, pero ahora ya se está trabajando en el proyecto y el estudio de nuevos sistemas de a bordo.

Italia participará con tres veleros en la Copa América de 1987.

El primero lo presenta el Yatch Club Costa Smeralda, con el "Consorzio Azzurra, Sfida Italiana America's Cup". El segundo el Yatch Club Italiano, con el Consorzio Italia, que ha adquirido el Victory 83 para utilizarlo como "liebre" para el nuevo 12 metros que proyectará y construirá el Consorzio (la nave liebre se utiliza para entrenar otra). El reglamento de la Copa prevé efectivamente que el velero y los aparejos deben ser diseñados y contruidos por la nación participante.

El último velero es el del Club Nautico Marina di Camarra, con el Consorzio Futura.

Alrededor de estos tres veleros están fermentando nuevas ideas y proyectos, sobre los cuales se mantiene obviamente la máxima reserva. Sin embargo, se asistirá a una potenciación del sistema de a bordo instalado en los 12 metros y a un desarrollo de los estudios meteorológicos del campo de la regata (análisis estadístico de la marcha de los vientos y de otros datos, con el auxilio de los satélites meteorológicos tipo Me-teosat), al desarrollo de nuevos sensores para el análisis de nuevos datos y, en general, a la adquisición de una mayor fiabilidad con los sistemas de proceso por parte de los componentes de la tripulación, para aprovechar las posibilidades al máximo.

Bruno Liotta

Instrucciones Basic para la gestión de las ventanas vídeo

En algunas máquinas existen implantaciones del Basic estándar que prevén las instrucciones necesarias para la gestión de las ventanas vídeo. Las modalidades operativas que se siguen en el uso de estas instrucciones comprenden dos fases: en la primera se genera la ventana definiendo sus dimensiones y su posición; después pueden direccionarse gráficos o leyendas haciendo referencia al nombre simbólico asociado a la ventana.

La definición de una ventana se realiza con instrucciones estructuralmente similares a la WINDOW construida en el ejemplo anterior, con la diferencia de que, en este caso, todas las subrutinas necesarias para la decodificación de la instrucción y para su ejecución forman parte del lenguaje, y son completamente transparentes para el usuario. En el ejemplo presentado, las subrutinas para la decodificación estaban escritas en Basic, y generaban una serie de llamadas a otras rutinas, siempre en Basic, que para ser ejecutadas antes debían ser traducidas por el intérprete.

Naturalmente, la sintaxis de las instrucciones de gestión de las ventanas puede variar de máquina a máquina. A continuación haremos referencia a la forma utilizada en el Olivetti M20.

En esta máquina, la pantalla vídeo prevé dos modalidades de presentación: 512 x 256 pixels o 480 x 256 pixels (la elección se hace en la fase de personalización del sistema). En los dos casos es posible utilizar toda la pantalla para trazar dibujos o en modalidad de texto, y una modalidad excluye la otra. En toda la pantalla, o en una de sus ventanas, cada punto de pantalla (pixel) está identificado bien por el sistema de coordenadas de base con origen abajo a la izquierda, bien por cualquier sistema de referencia definido por el usuario.

Sin embargo, en este segundo caso puede suceder que las coordenadas de un punto no correspondan a ningún pixel, puesto que la posición de cada pixel está definida por el hardware; en este caso, el sistema activa el pixel más cercano a las coordenadas indicadas.

Este modo de funcionamiento desvincula de la necesidad de utilizar referencias congruentes con los puntos de pantalla, pero pueden producirse imprecisiones también notables en la presentación de las imágenes gráficas.

La instrucción que permite definir una ventana tiene la forma

$$A = WINDOW (q,p,v,h)$$

donde

A es la variable (entera) asociada a la ventana. El sistema le asigna un valor progresivo de 2 a 16, puesto que 16 es el máximo número de ventanas que pueden abrirse simultáneamente, mientras que el valor 1 indica la pantalla completa. La numeración progresiva empieza por el primer valor disponible y avanza a medida que se abren nuevas ventanas, pero con la reutilización de los números dejados libres por un eventual cierre de ventanas anteriores. Por ejemplo, si se han creado las ventanas 2, 3 y 4, la próxima será la 5, a menos que se haya borrado (o sea cerrado) una de las anteriores, en cuyo caso, la próxima ventana, en lugar de asumir el valor 5, adquirirá el que ha quedado libre.

q Especifica la posición ocupada por la ventana que se crea en el interior de la ventana «generatriz». Esta función es algo diferente de la correspondiente utilizada en el ejemplo anterior, en el que la ventana sólo podía posicionarse relativamente en el interior del vídeo. Y viceversa, en esta máquina, cada ventana puede utilizarse integralmente como una pantalla vídeo (toda la pantalla es la ventana 1) y en su interior pueden posicionarse otras ventanas. Los valores introducidos por el parámetro q son:

- 0 la nueva ventana se posiciona en la parte alta de la ventana generatriz
- 1 en la parte baja
- 2 a la izquierda
- 3 a la derecha

p Indica la posición en correspondencia con la cual la ventana generatriz (eventualmente toda la pantalla) se divide para originar las nuevas ventanas.

El valor a asignar al parámetro p depende del de q. Si q vale 0 o 1 (ventana situada en la parte alta o en la baja de la generatriz), para crear una nueva ventana, el valor de p puede variar entre 1 y 255, e indica el número de líneas de barrido que

constituyen las ventanas que se crean*. Abajo se ha representado un ejemplo de subdivisión de la pantalla.

Si el parámetro q (cuadrante) vale 2 o 3, la subdivisión es vertical, y el parámetro p (posición) puede variar entre 1 y la longitud de la ventana generatriz menos 1.

v Es un parámetro opcional que indica el número de líneas de barrido para cada línea de texto. Puede variar entre 10 y 16. Si se omite, se asume el de la ventana generatriz.

h Establece la distancia, en pixels, entre dos caracteres de texto. Puede asumir los valores 6 u 8; el primero permite un máximo de 80 caracteres en cada línea, y el

* Con el término línea de barrido se indica una línea de la pantalla en modo gráfico. El término utilizado no debe confundirse con el término línea, que se refiere a la modalidad de texto. En este caso, una línea dista de las otras por lo menos la altura de un carácter, y está compuesta por un cierto número de líneas de barrido, generalmente de 7 a 12.

segundo 64. Es opcional, y por omisión es igual al de la ventana generatriz.

En la página siguiente se han indicado algunas instrucciones WINDOW que muestran la influencia de los parámetros v y h. Los otros parámetros se han omitido y, por tanto, el sistema asume los de la ventana generatriz.

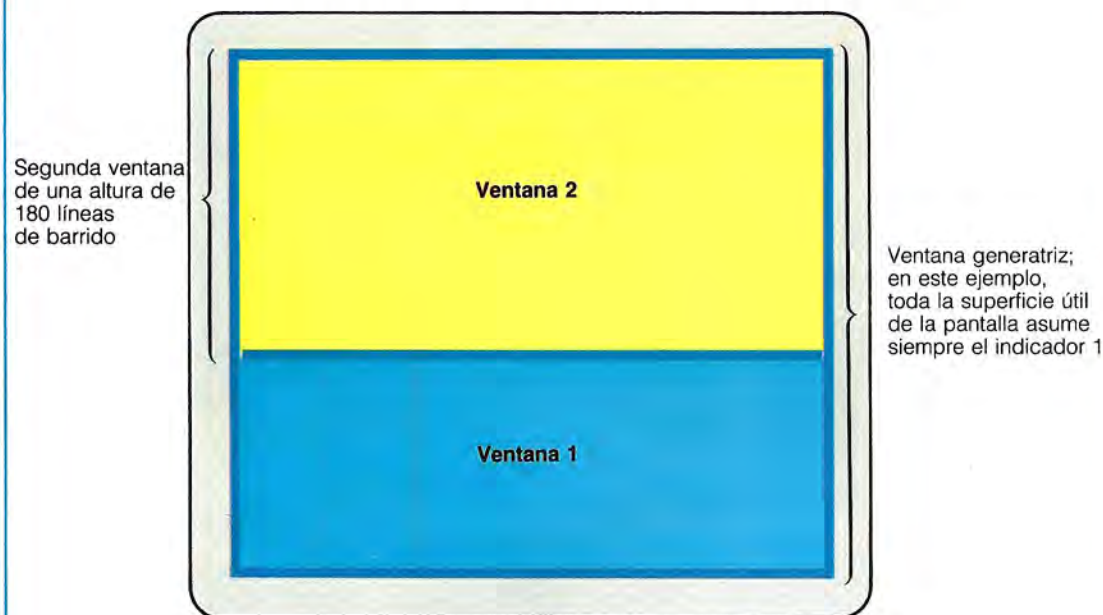
Los parámetros v y h pueden variarse en cualquier momento (incluso después de haber abierto la ventana) son la misma instrucción WINDOW, utilizada en la forma

$$A = \text{WINDOW}(0,0,v,h)$$

En este caso, la variable A es ficticia, puesto que el sistema procede a asignarle el valor correspondiente a la ventana seleccionada. La selección de una ventana también se produce con la instrucción WINDOW, puesta en la forma

$$\text{WINDOW \%N}$$

EJEMPLO DE SUBDIVISION DE LA PANTALLA EN FUNCION DE LOS PARAMETROS q Y p



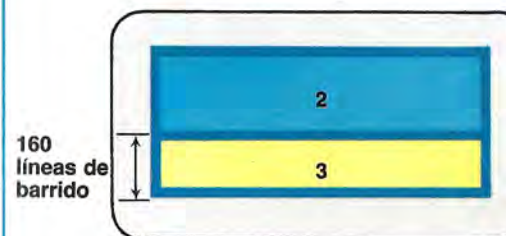
Abriendo una ventana (por ejemplo la número 2) con los valores

q = 0 la ventana se posiciona en la parte alta de la generatriz (en este caso toda la pantalla) y la división es horizontal

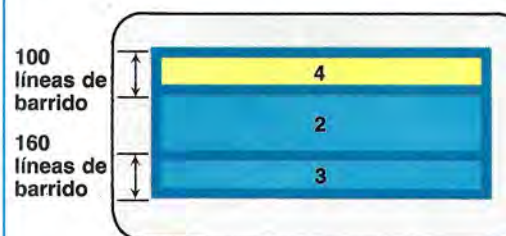
p = 180 la altura de la ventana, a partir del extremo superior, es de 180 líneas de barrido

Ventana generatriz; en este ejemplo, toda la superficie útil de la pantalla asume siempre el indicador 1

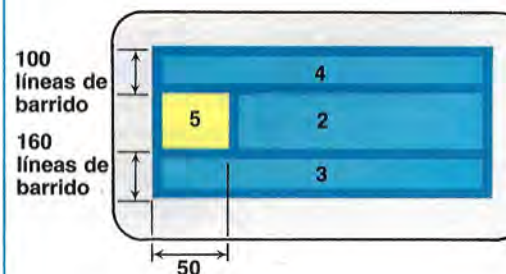
EJEMPLOS DE USO DE LA INSTRUCCION WINDOW



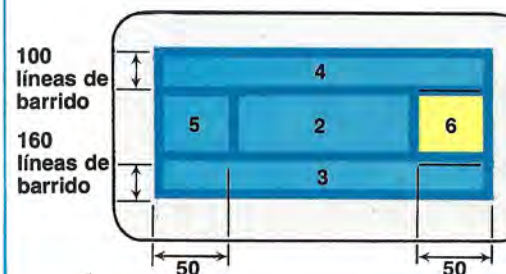
WINDOW (1,160). Subdivisión en la parte baja (q = 1) de 160 líneas de barrido de altura



WINDOW (0,100). Subdivisión en la parte alta de 100 líneas de barrido de altura



WINDOW (2,50). Subdivisión en la parte izquierda. En este caso, el valor 50 del parámetro p indica la anchura de la ventana



WINDOW (3,50). Análoga subdivisión realizada a la derecha (q = 3) de la ventana generatriz

que selecciona la ventana correspondiente al valor numérico contenido en la variable interna N. Por ejemplo, con las instrucciones

$$N = 2 \\ \text{WINDOW \%N}$$

se selecciona la ventana número 2. El valor nu-

mérico también puede proporcionarse directamente, y las dos instrucciones anteriores pueden quedar reunidas en la instrucción única WINDOW %2.

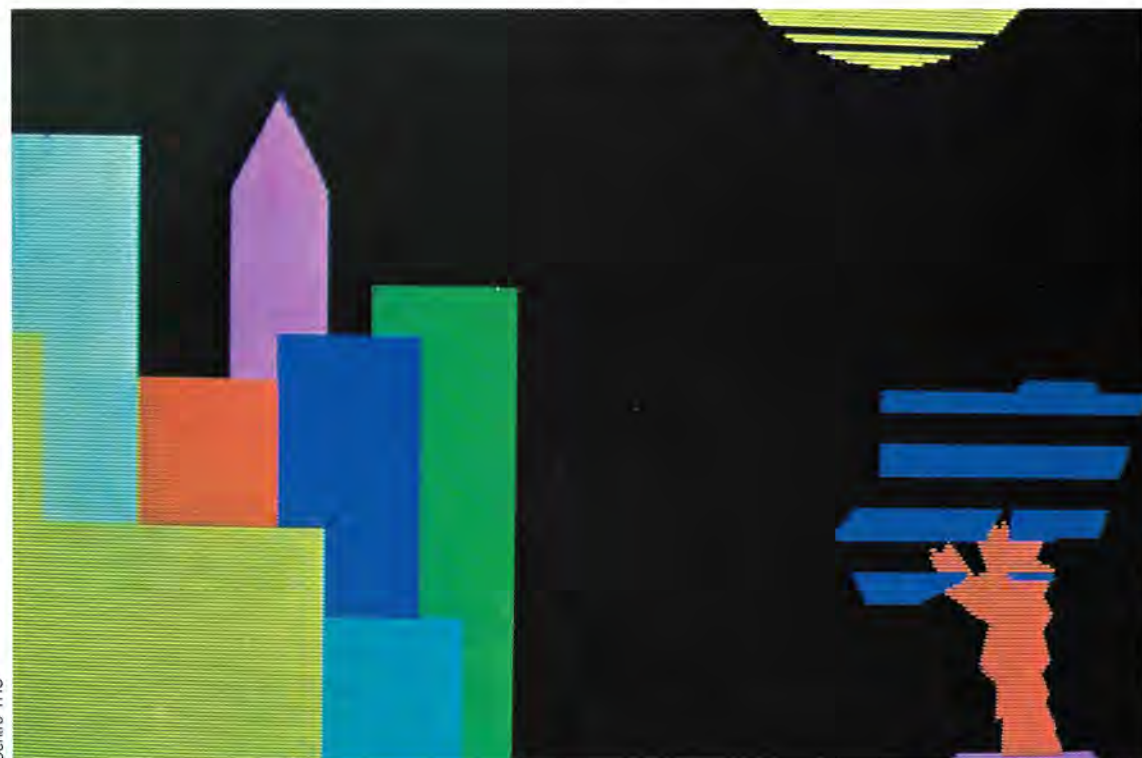
La instrucción WINDOW también se adopta para cerrar una ventana que ya no es necesaria, dejando así disponible el espacio para otras aplicaciones. La forma es

donde A indica la ventana a cerrar. Omitiendo el parámetro A se cierran todas las ventanas presentes en la pantalla.

Vectores gráficos y tabla de las figuras

Un dibujo cualquiera puede descomponerse en una serie de trazos lineales elementales. Utilizando esta técnica, en algunos sistemas es posible definir un conjunto de desplazamientos elementales de la punta de escritura que sirven para representar la figura. En general, estos conjuntos se indican con el término de **tabla de las figuras**, y tienen unas funcionalidades muy similares a las utilizadas en la tabla de los desplazamientos para generar números o letras. En los sistemas en que se utilizan, las tablas de las figuras son gestionadas por el software de base, que por tanto debe prever instrucciones para el cambio de escala o para la rotación. En las máquinas en las que esto no está previsto, siempre es posible crear un software muy sencillo que realice las mismas funciones.

Una ciudad "interpretada" por el ordenador en modalidad gráfica.



Codificación de los vectores de desplazamiento

La generación de una tabla de las figuras consiste en memorizar en la escala oportuna todos los desplazamientos necesarios para definir el dibujo. Generalmente, los tipos de desplazamiento previstos son cuatro (arriba, abajo, derecha, izquierda).

Una figura de la tabla no se memoriza en dimensiones reales, sino que se recurre a una normalización realizada suponiendo un desplazamiento igual a la medida del lado más corto. Por ejemplo, si hay que memorizar un rectángulo de altura doble que la base, la tabla resultante será del tipo:

- 1 / un desplazamiento horizontal (base)
- 2 / dos desplazamientos verticales (altura = 2 x base)
- 3 / un desplazamiento horizontal (en sentido opuesto al anterior)
- 4 / dos desplazamientos verticales (en sentido opuesto a los anteriores)

Utilizando, para indicar los desplazamientos, la simbología que sigue

- ↑ = desplazamiento unitario vertical negativo
- ↓ = desplazamiento vertical positivo
- = desplazamiento horizontal positivo
- ← = desplazamiento horizontal negativo

el rectángulo del ejemplo se identificará con la sucesión

→ ↑ ↑ ← ↓ ↓

Cada desplazamiento se llama **vector gráfico**, y el rectángulo anterior se dirá que está constituido por 6 vectores gráficos. En la fase de reconstrucción del dibujo, cada uno de los vectores gráficos así definidos indica la dirección en que debe moverse para trazar el correspondiente lado, mientras que la amplitud del desplazamiento está definida por el factor de escala. De esta manera es muy sencillo efectuar cambios de escala: sólo debe reconstruirse el dibujo realizando los desplazamientos según las mismas direcciones indicadas por los vectores gráficos, pero en cantidades múltiples en función de la nueva escala.

Esquematisando cada figura como constituida por una combinación de cuatro desplazamientos elementales, puede obtenerse la memorización con notable ahorro de espacio. Las posibles elecciones para cada paso sólo son cuatro (↑, →, ←, ↓) y pueden ser representadas utilizando sólo dos bits, por ejemplo poniendo

Desplazamiento	Valor decimal	Valor binario
↑	0	00
→	1	01
↓	2	10
←	3	11

La codificación representada es la más frecuente en las máquinas que utilizan esta metodología (Apple y compatibles). Aunque en otros casos pueden tenerse códigos diferentes (como valor numérico), los vectores siempre pueden representarse con dos bits.

La tabla muestra una posible codificación para representar los vectores gráficos, pero no indica los eventuales atributos de cada vector. En particular, por lo menos deberá especificarse si el desplazamiento indicado por el vector sólo es un posicionado y, por tanto, sin que aparezca en la pantalla, o bien se trata de un segmento en visión. En el primer caso (posicionado), el desplazamiento descrito por el vector debe trazarse con un color igual al del fondo y, en el segundo caso, con un color visible. Para proporcionar esta indicación es necesario utilizar un tercer bit que indica si el segmento está en visión o no. Si se asocia el tercer bit el valor 0 para indicar un simple desplazamiento y el valor 1 para indicar un trazo visto, se tienen las correspondencias indicadas en la tabla al pie de esta página.

Como puede verse, para pasar de un desplazamiento no visto al mismo en visión, basta con sumar 4. Por ejemplo, el desplazamiento horizontal positivo (→) tiene código decimal 1 si no está en visión, mientras que es 5 si está en visión. Con esta codificación, el rectángulo considerado anteriormente está representado por los valores decimales 5 (→), 4 (↑), 4 (↑), 7 (←), 6 (↓), y por los valores binarios:

- 101 (5, →)
- 100 (4, ↑)
- 100 (4, ↑)
- 111 (7, ←)
- 110 (6, ↓)
- 110 (6, ↓)

La memorización de esta tabla requeriría 6 posi-

CODIFICACION DE LOS VECTORES DE DESPLAZAMIENTO

Desplazamiento (vector)	En visión			No en visión			Valor decimal en visión	Valor decimal no en visión
	2	1	0	2	1	0		
↑	1	0	0	0	0	0	4	0
→	1	0	1	0	0	1	5	1
↓	1	1	0	0	1	0	6	2
←	1	1	1	0	1	1	7	3

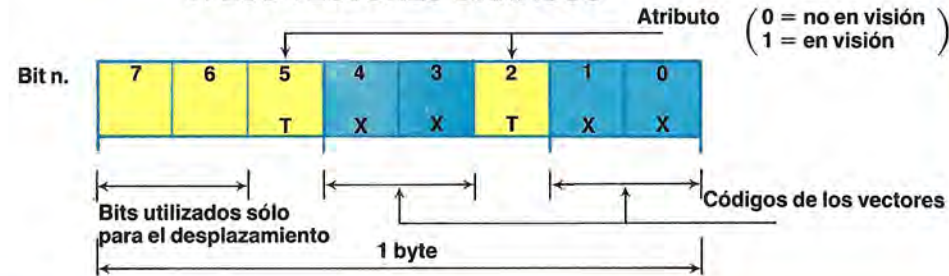
ciones de memoria, una para cada desplazamiento. Sin embargo, procediendo de esta manera, se tendría un despilfarro inútil, puesto que cada posición está compuesta por 8 bits, mientras que un vector gráfico sólo utiliza 3. Por tanto, es natural buscar una fórmula de compactación que permita memorizar más vectores en la misma posición.

La más utilizada, indicada en la primera de las dos figuras de abajo, consiste en ocupar con dos vectores sucesivos 6 de los 8 bits de una posición de memoria: los bits de 3 a 5 para el primero y los de 0 a 2 para el segundo. En la posición de memoria quedan dos bits inactivos (el 6 y el 7), que en algunos casos pueden utilizarse para indicar un vector de sólo desplazamiento (falta el bit que indica el atributo). Sin embargo, su empleo no puede ser generalizado y, por tanto, suele evitarse su uso.

En la figura del pie de esta página se ha representado una posible forma de memorización de un rectángulo obtenida aprovechando la compactación descrita anteriormente. Los bits 6 y 7 están siempre a 0, mientras que el 5 y el 2 están

siempre a 1 (indicando desplazamiento en visión). Para indicar a la máquina del final de la tabla se pone a 0 una posición de memoria (1 a 4). Una vez memorizada la tabla de la figura, para obtener el dibujo hace falta una rutina que analice el contenido de cada memoria y trace un segmento en la dirección indicada por los bits 3, 4 y 0, 1 en el color indicado por los bits 5 y 2. El bucle debe interrumpirse cuando todo el contenido de la memoria es cero, valor con el cual se ha convenido indicar el final de la tabla. En las figuras de las páginas 1593 a 1595 se ha representado el diagrama de flujo de un programa de demostración y, en la página 1596, el listado correspondiente. En este caso de ejemplo, la tabla de la figura (rectángulo) es fija, introducida a través de un DATA, mientras que la fase de representación (rutina 1000) puede utilizarse para empleos generales. El programa mostrado sólo es un ejemplo válido desde el punto de vista cualitativo. Le faltan dos funciones esenciales: la posibilidad de incluir más figuras en la misma tabla y las subrutinas para la generación y la memorización de las figuras.

COMPACTACION DE LOS CODIGOS CORRESPONDIENTES A LOS VECTORES GRAFICOS



MEMORIZACION DE UN RECTANGULO

Posición de memoria	7	6	32 5	16 4	8 3	4 2	2 1	1 0	Decimal
1	0	0	1	0	1	1	0	0	44
2	0	0	1	0	0	1	1	1	39
3	0	0	1	0	1	1	0	1	45
4	0	0	0	0	0	0	0	0	0

5(→) + 4(↑)
4(↑) + 7(←)
6(↓) + 6(↓)

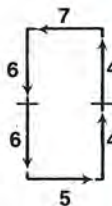
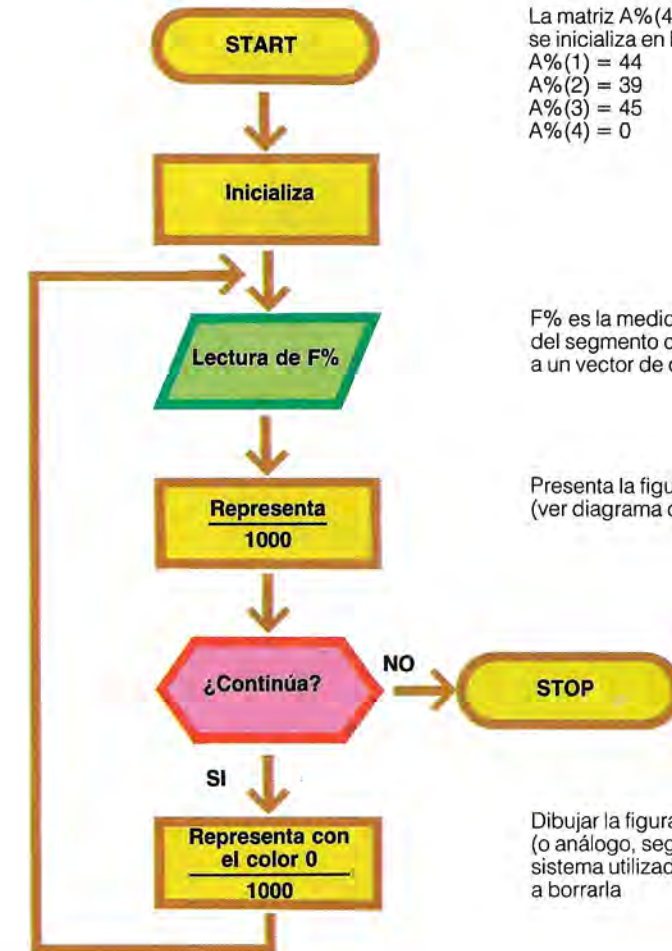


DIAGRAMA PARA LA PRESENTACION DE UNA TABLA DE LAS FIGURAS



La matriz A%(4), tabla de la figura, se inicializa en los valores:
A%(1) = 44
A%(2) = 39
A%(3) = 45
A%(4) = 0

F% es la medida, en puntos de pantalla, del segmento correspondiente a un vector de desplazamiento

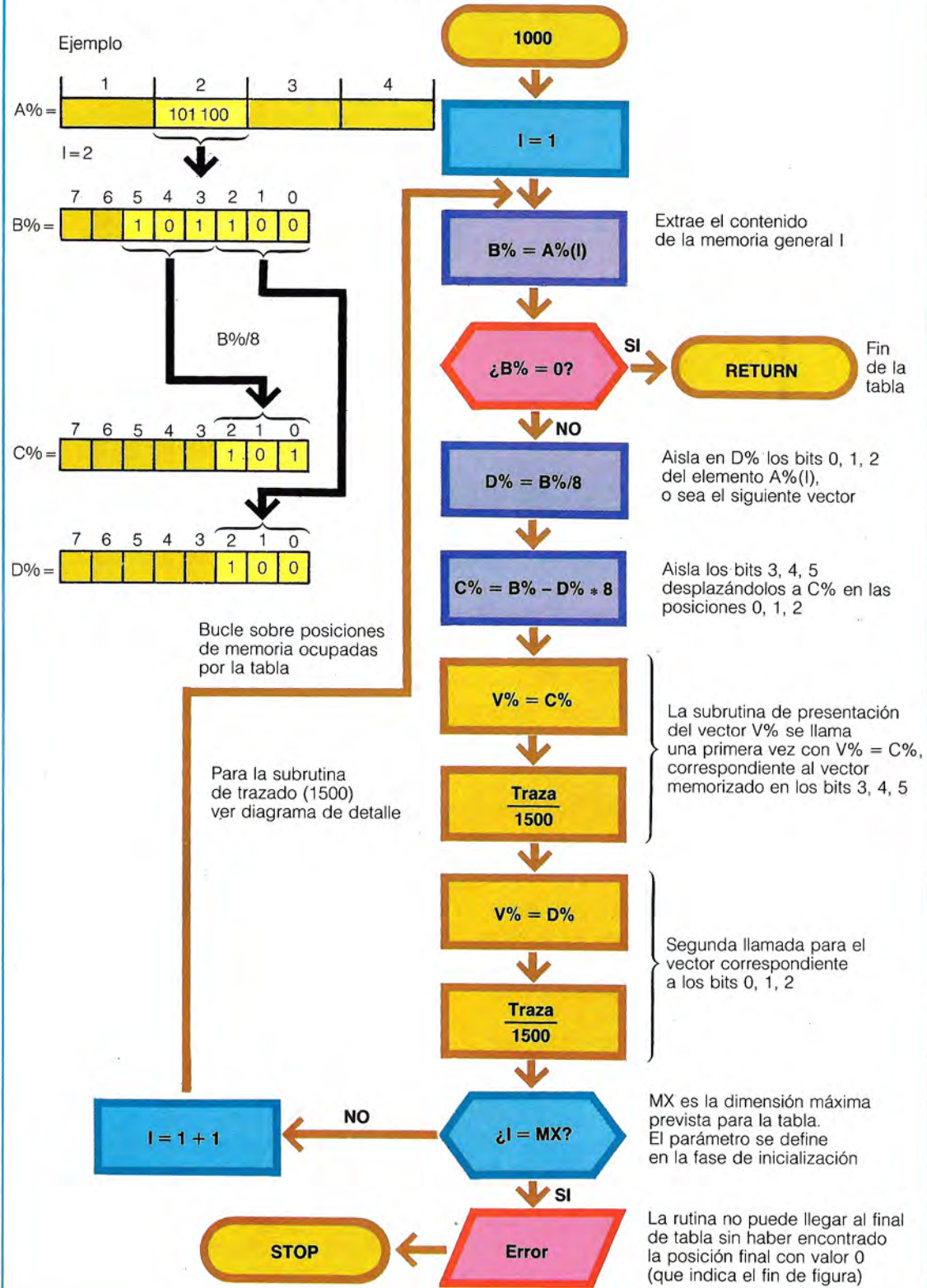
Presenta la figura (ver diagrama de detalle)

Dibujar la figura con color 0 (o análogo, según el sistema utilizado) equivale a borrarla

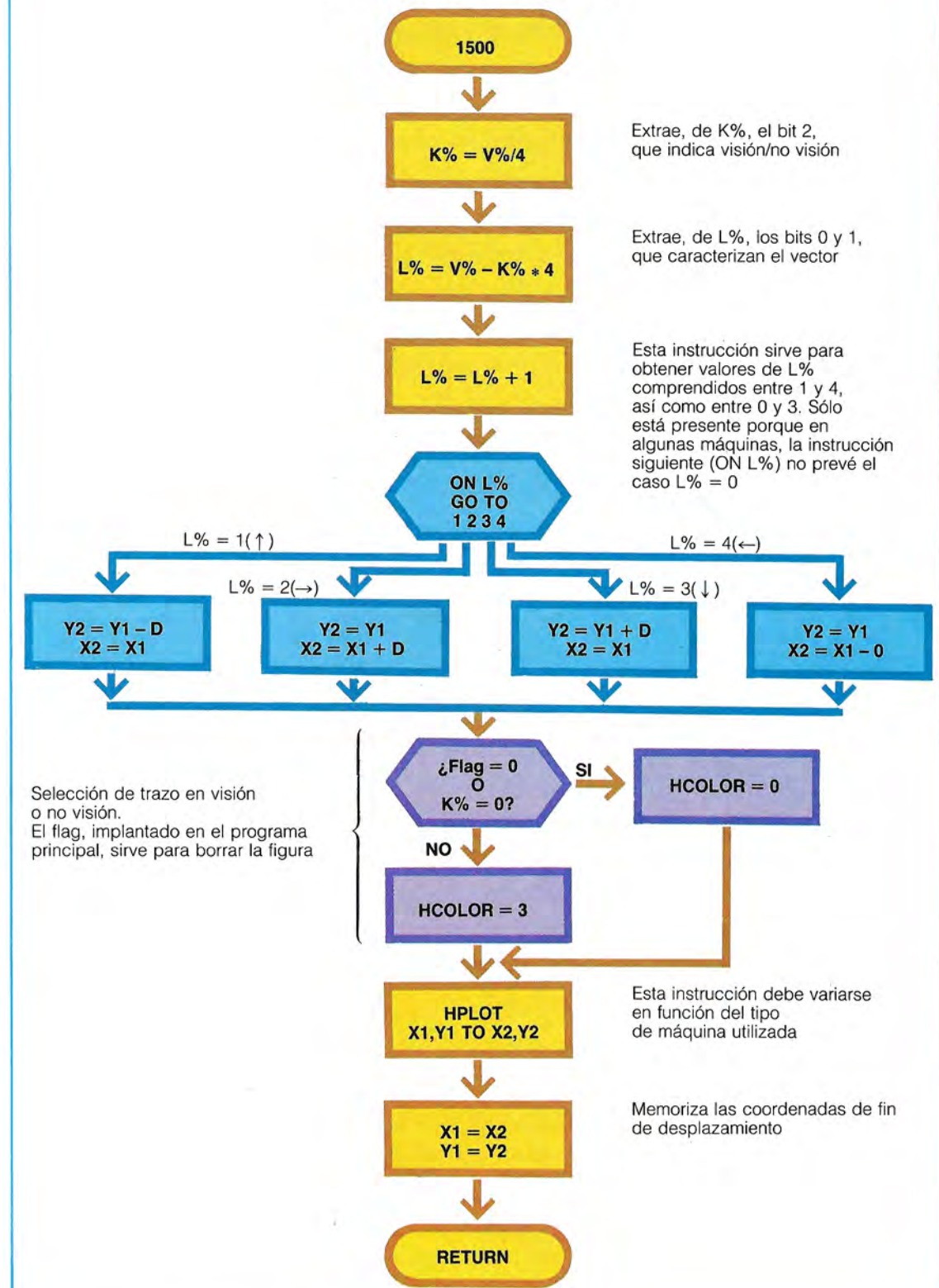
En los empleos prácticos no es conveniente construir un dibujo complejo en base a una serie única de símbolos. El mejor enfoque consiste en dividir un cierto número de figuras sencillas (**shapes**) y en memorizar el conjunto de una única tabla de las figuras (**shape table**). En este caso, la estructura que debe darse a la tabla es más compleja, puesto que debe contener una serie de punteros que indiquen de cuántas figuras está compuesto el dibujo y en qué punto de la tabla empieza cada una de ellas. En la página 1597 se muestra una estructura típica utilizada en las máquinas que tienen este tipo de software. La primera posición (byte 0) contiene el número de las figuras que hay en la tabla; la segunda siempre es nula. Las que quedan están

divididas en dos grupos: el primero es la zona de las direcciones, que contiene los punteros al principio de cada figura, y constituye el directorio de la tabla. El segundo grupo es el conjunto de los vectores gráficos que representan cada figura de la tabla. El final de cada figura se indica con una posición de contenido nulo. El sistema examina el contenido del byte 0 (primera posición) y así se entera del número de figuras presentes (en el ejemplo, 3). Como el segundo byte es siempre cero, debe saltarse, y las direcciones de cada figura, que son 3, se encontrarán a partir de la posición 2 y procediendo con paso 2. Es decir, las posiciones 2 y 3 contendrán el puntero al principio de la primera figura, las 4 y 5 el puntero a la segunda y las 6

SUBROUTINA DE PRESENTACION



SUBROUTINA DE TRAZADO



GESTION DE UNA TABLA DE LAS FIGURAS

Versión Siprel, Apple y compatibles

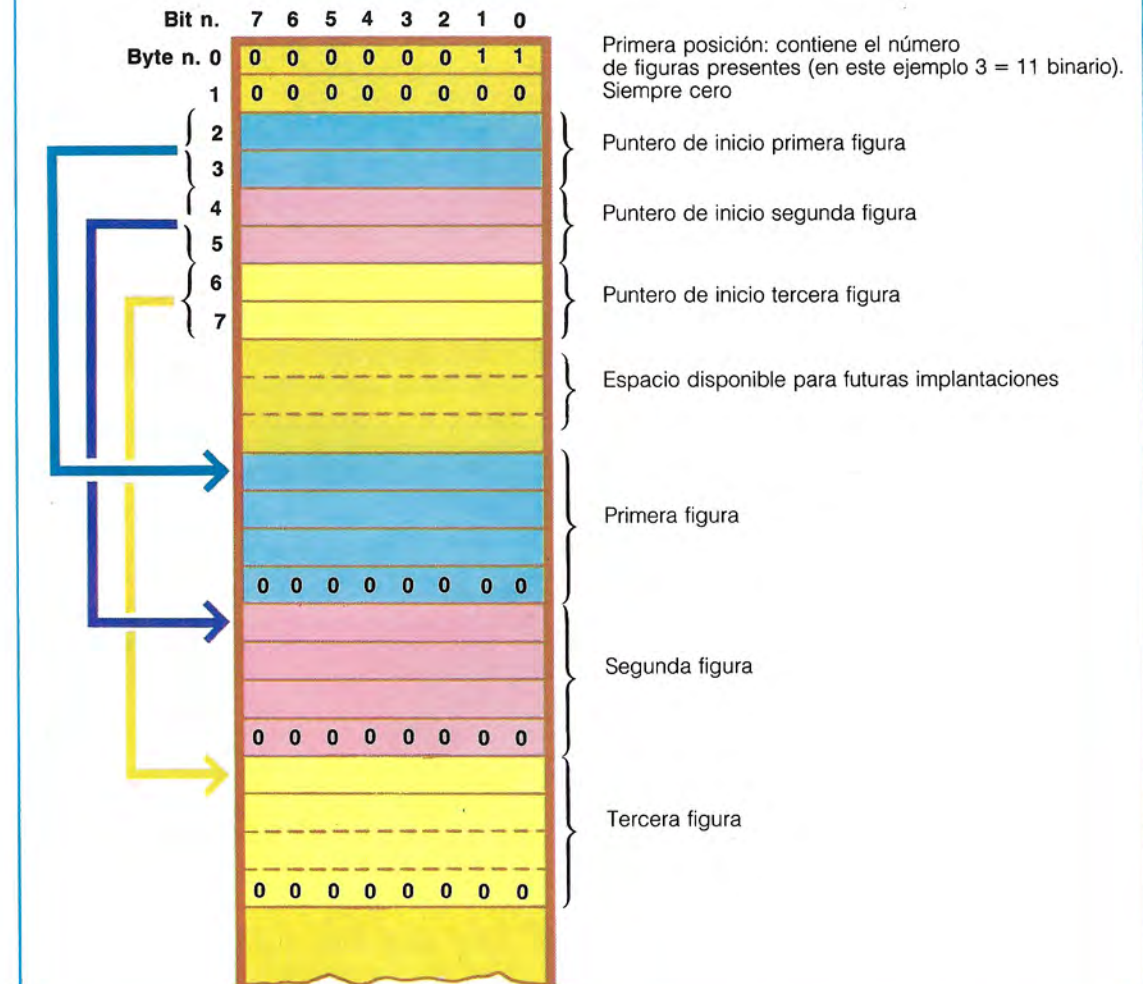
```

1 REM =====
2 REM * TABLA *
3 REM * FIGURA *
4 REM =====
5 :
6 :
50 HGR
100 DIM A%(20) : MX = 20
103 :
106 REM -----
107 REM :CARGA VECTORES:
108 :
110 A%(1) = 36:A%(2) = 55:A%(3) =
    46
112 REM -----
115 :
120 X1 = 100:Y1 = 50:X2 = X1:Y2 =
    Y1
125 FL = 1
150 HOME : VTAB 22: INPUT "LONGI
    TUD SEGMENTO ? ";F%
155 D = F%
160 GOSUB 1000
170 HOME : VTAB 22: INPUT "CONTI
    NUD ? ";A$
180 IF LEFT$(A$,1) = "S" THEN
    200
190 TEXT : HOME : END
200 FL = 0:X1 = 100:Y1 = 50:X2 =
    X1:Y2 = Y1
210 GOSUB 1000
220 GOTO 125
999 :
1000 REM =====
1001 REM * TRAZA GRAFICO *
1002 REM =====
1003 :
1010 FOR I = 1 TO MX
1020 B% = A%(I)
1030 IF B% = 0 THEN I = MX: NEXT
    I: RETURN
1040 D% = B% / 8:C% = B% - D% * 8

1050 V% = C%: GOSUB 1500
1060 V% = D%: GOSUB 1500
1070 NEXT I
1080 HOME : VTAB 22: PRINT "ERRO
    R !": RETURN
1499 :
1500 REM =====
1501 REM * TRAZA *
1502 REM =====
1503 :
1510 L% = V% / 4:L% = V% - K% * 4

1520 L% = L% + 1
1530 ON L% GOTO 1550,1560,1570,1
    580
1550 Y2 = Y1 - D:X2 = X1: GOTO 16
    00
1560 Y2 = Y1:X2 = X1 + D: GOTO 16
    00
1570 Y2 = Y1 + D:X2 = X1: GOTO 16
    00
1580 Y2 = Y1:X2 = X1 - D
1600 IF FL = 0 OR K% = 0 THEN HCOLOR= 0: GOTO 1620
1610 HCOLOR= 3
1620 HPLOT X1,Y1 TO X2,Y2
1630 X1 = X2:Y1 = Y2: RETURN
    
```

ESTRUCTURA TIPICA DE UNA TABLA DE LAS FIGURAS



y 7 a la tercera. Generalmente, entre el principio de las figuras y las tablas se deja una posición implantada a cero para poder insertar nuevos punteros si las dimensiones y la complicación de dibujo tuviesen que aumentar. Con esta estructura resulta claro que la subrutina de creación de una tabla de las figuras no debe limitarse sólo a memorizar los vectores, sino que también debe preparar el directorio.

Programa para la generación y la gestión de tablas de las figuras

Los métodos de creación de una tabla de las

figuras son fundamentalmente dos. El primero, más sencillo pero menos útil, prevé un coloquio a través del cual el usuario puede indicar los desplazamientos a realizar. El segundo, más complejo desde el punto de vista de la programación, pero mucho más útil, se activa asociando cuatro teclas a cuatro desplazamientos fundamentales; cada desplazamiento se presenta y se memoriza en la sucesión en que es activado. En términos de la generación de la tabla, el resultado es el mismo, pero el usuario ve formarse el dibujo mientras lo introduce.

Ahora examinaremos un programa para la ge-

neración y para el empleo de tablas de las figuras utilizables en los sistemas que no prevén estas funciones. Con modificaciones oportunas, que como máximo se reducen a la eliminación de algunas partes, el mismo programa puede ser utilizado también en los sistemas que ya prevén la gestión de las tablas de los desplazamientos.

Las principales funciones que debe realizar el programa son las siguientes:

- Adquisición de las figuras que constituyen el dibujo, su memorización en forma de desplazamientos y presentación en la pantalla
- Transferencia de la tabla de los desplazamientos al diskette, con posibilidad de relectura para sucesivas modificaciones
- Posibilidad de variar la escala, desplazar o rotar cada figura simple.

En la figura de abajo se ha representado el diagrama de flujo del main y en la tabla de la página siguiente las variables utilizadas. Las subrutinas principales (a nivel de bloque funcional) sólo son cuatro.

La primera (1000) se utiliza para inicializar las variables, los DATA y los flags. Sigue la lectura de un carácter (A\$) realizada en el main, que debe ir seguida de la instrucción INPUT\$(1) (bajo CP/M) o con la GET (o equivalentes), que no tienen eco y que permiten introducir un solo carácter sin terminador de record (en la INPUT normal es necesario cerrar el record de introducción pulsando la tecla RETURN).

El carácter así introducido se analiza en la subrutina 2000, que responde con tres valores, contenidos respectivamente en K, TS y TP. El valor de K, que puede estar comprendido entre 1 y 3, indica el tipo de tecla pulsada:

VARIABLES Y SIMBOLOS UTILIZADOS EN EL PROGRAMA

A\$	=	Carácter en lectura. Puede indicar un desplazamiento o un comando
A1\$	=	Segundo carácter de complemento del comando
NM\$	=	Nombre del file en memorización o recuperación figura
K	=	Indicador del tipo de tecla pulsada (no válido, desplazamiento, orden)
TS	=	Tipo de desplazamiento deseado
TP	=	Tipo de comando
S(10)	=	Matriz que contiene los códigos ASCII de las teclas de desplazamiento
O(10)	=	Matriz que contiene los códigos ASCII de las teclas de comando
X1, Y1	=	Coordenadas antes del desplazamiento
X2, Y2	=	Coordenadas después del desplazamiento
DX, DY	=	Valores del desplazamiento unitario (puntos de pantalla)
HC	=	Flag que indica desplazamiento en visión o no en visión. Se activa con TP = 1 ó 2
B(100)	=	Matriz que contiene los desplazamientos que constituyen la figura
IN	=	Índice de la última posición ocupada en B(100)
X0, Y0	=	Coordenadas de principio de dibujo
FL	=	Flag de desplazamiento
FR	=	Flag de rotación
FS	=	Flag de escala

Teclas comando	D = Disco (salva)
S = Desplaza	C = Borra
R = Rota	+ = Amplía escala
L = Carga	- = Reduce escala

- 1 Tecla no prevista
- 2 Tecla de desplazamiento
- 3 Tecla de introducción de comandos

- 5 Memoriza la figura
- 6 Reclama una figura memorizada
- 7 Borra
- 8 Amplía
- 9 Reduce

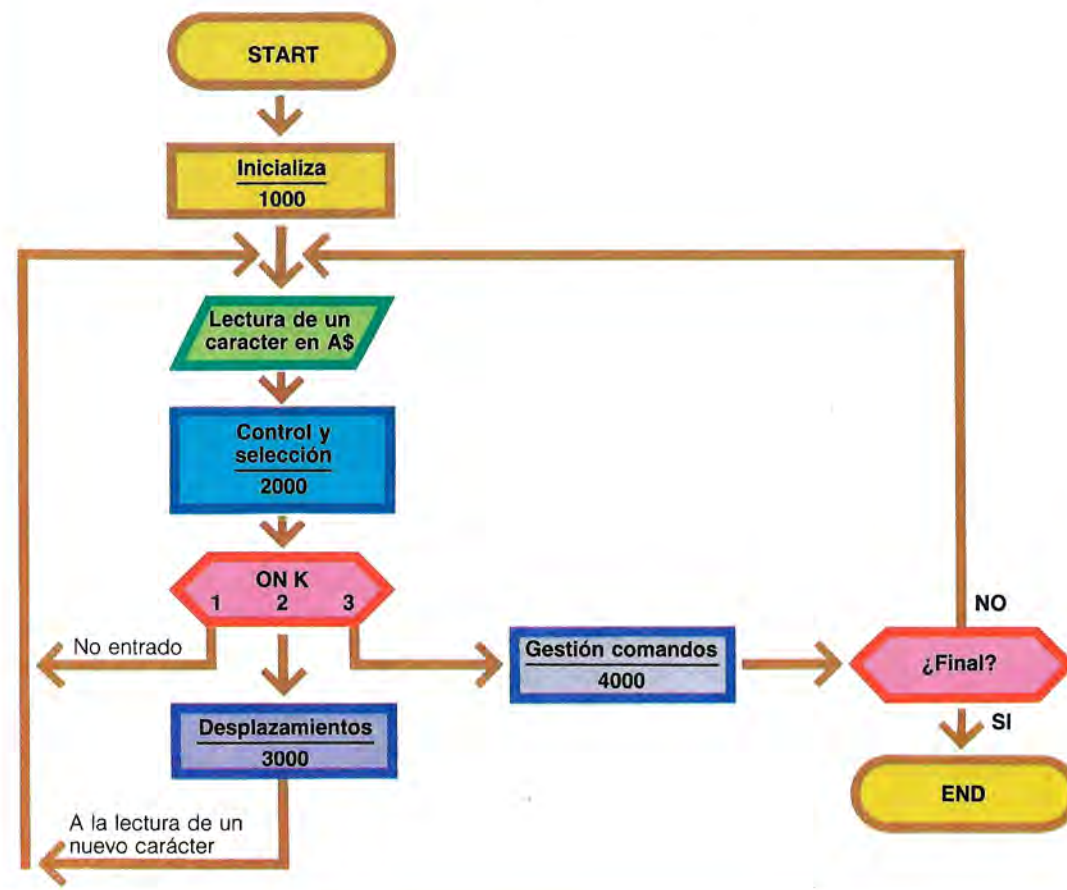
En el caso de desplazamiento (para trazar la figura) o de orden (memorización, cambio de escala, etc.), los valores asociados están en las variables TS (desplazamiento) y TP (orden). La variable TS puede asumir un valor comprendido entre 1 y 4 en función del desplazamiento deseado (↑, →, ↓, ←), mientras que TP puede tener valores comprendidos entre 1 y 9, con los siguientes significados:

- 1 Los desplazamientos que siguen a este comando no están en visión
- 2 Presenta los desplazamientos
- 3 Desplaza la figura en la dirección especificada por la tecla pulsada después
- 4 Rota la figura en el sentido indicado por la tecla que sigue

El diagrama de flujo de detalle de la subrutina 2000 puede verse en la página siguiente. Como parámetro de entrada es necesario pasar el carácter a A\$. La rutina busca si este carácter está comprendido entre los previstos para los desplazamientos (en la matriz S) o para las órdenes (matriz O). En caso afirmativo transfiere a las respectivas variables (TS o TP) la posición en que se ha encontrado el carácter. Este valor indica también el tipo de desplazamiento o de orden para actuar.

En la página 1601 se ha representado el diagrama de flujo de la subrutina 3000, antepuesta a la gestión de los desplazamientos. Las funciones realizadas, también activadas a través de la llamada a otras subrutinas, son las siguientes:

PROGRAMA PARA LA GENERACION Y LA GESTION DE TABLAS DE LAS FIGURAS



SUBROUTINA DE CONTROL SOBRE EL CARACTER INTRODUCIDO

K se inicializa al valor 1 para indicar carácter no introducido. Si en el curso de la rutina A\$ no es reconocido, este valor permanece inalterado e indica error al main

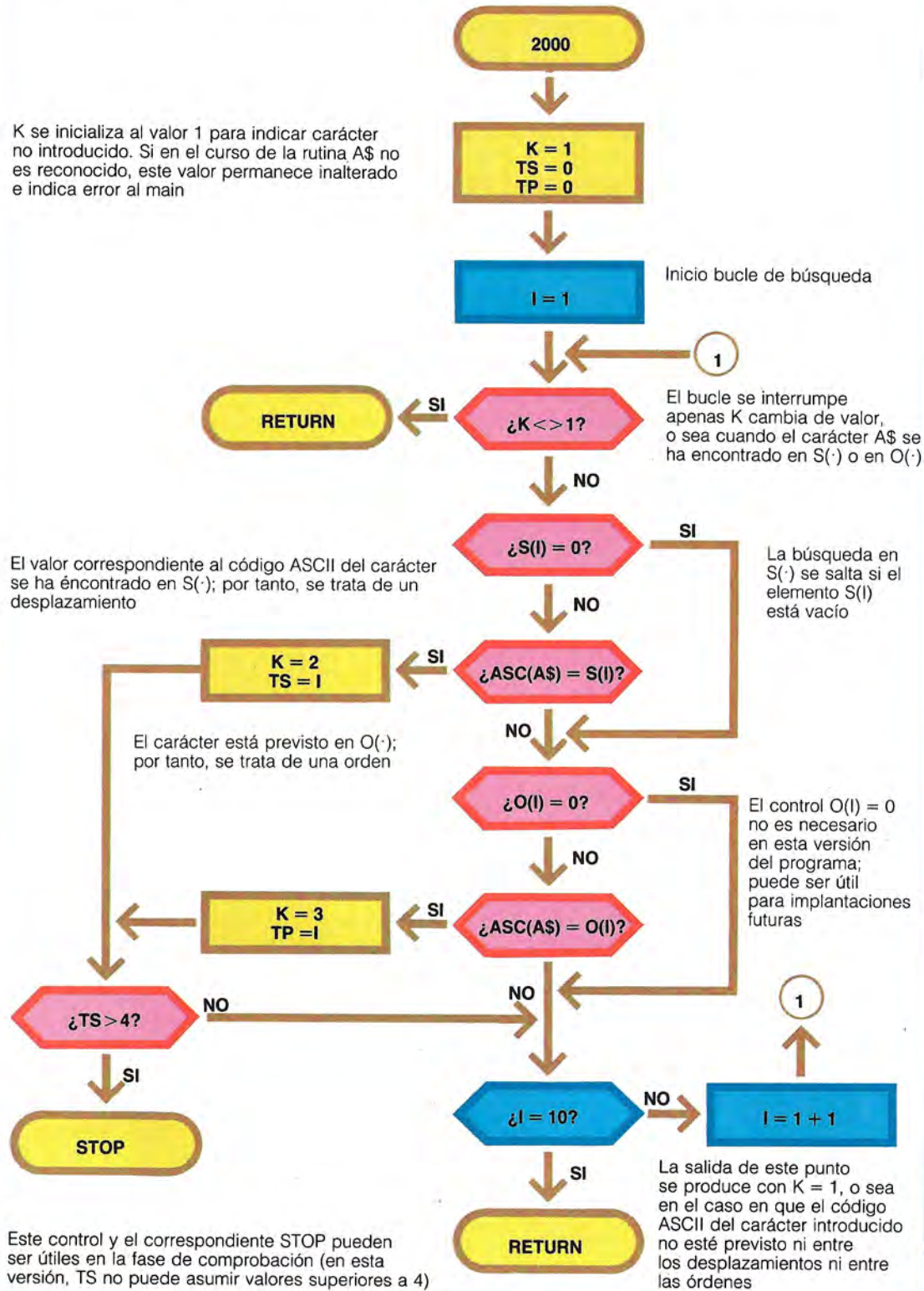
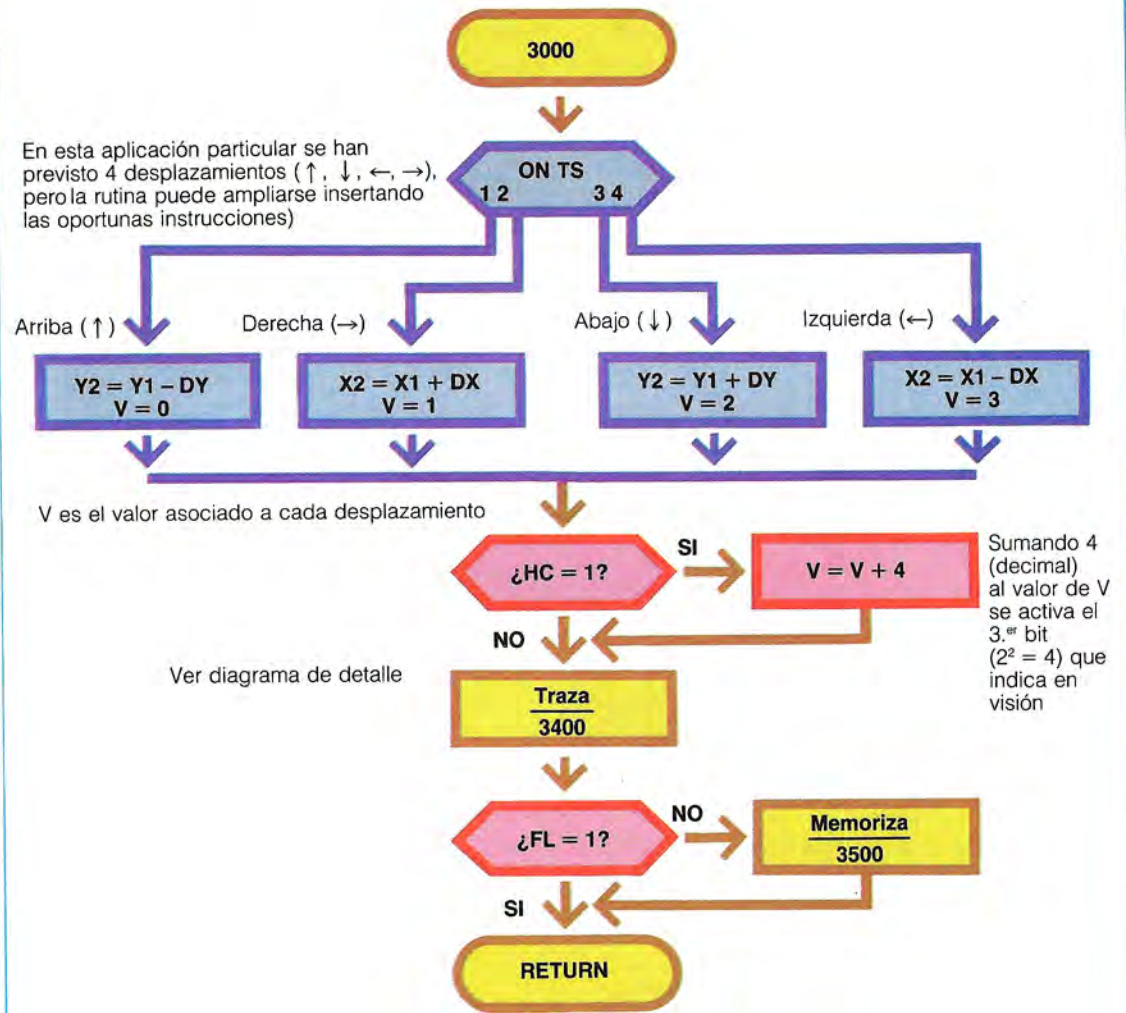


DIAGRAMA DE LA SUBROUTINA DE GESTION DE LOS DESPLAZAMIENTOS



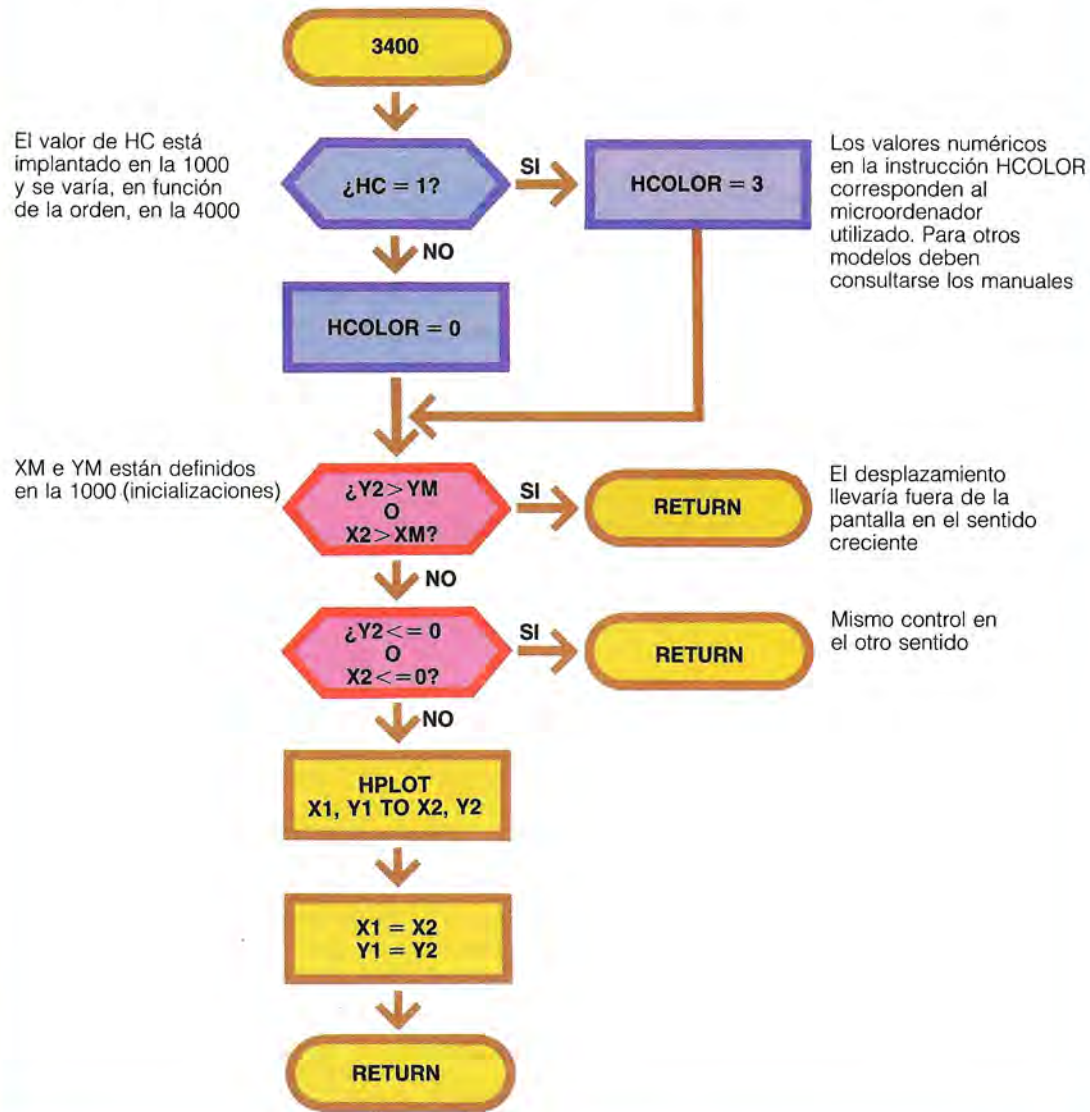
- Cálculo de las coordenadas después del desplazamiento y del valor V a memorizar en la tabla
- Transferencia del desplazamiento a la tabla (subrutina 3500)
- Trazado del desplazamiento (subrutina 3400).

Las nuevas coordenadas se calculan sumando a las coordenadas X1, Y1 los valores DX y DY; los primeros valores de X1 e Y1 deben implantarse en la subrutina 1000.

El valor de V a memorizar en la tabla de los desplazamientos sólo se ha presentado por motivos de claridad: podría obtenerse simplemente poniendo $V = K - 1$, o haciendo variar K entre 0 y 3. De esta manera no sería necesaria la nueva variable V (K cae en el margen de memorización, y podría utilizarse directamente), pero se tendrían complicaciones en el programa.

Obsérvese finalmente, que el cálculo de Y2 está adaptado a las máquinas que tienen el origen del vídeo arriba a la izquierda; en cambio, para la otra orientación deben invertirse los signos.

SUBROUTINA DE TRAZADO DEL DESPLAZAMIENTO ELEMENTAL



Las condiciones de desplazamiento en visión o no en visión se comprueban con el flag CH, que asume el valor 1 cuando se introduce el carácter correspondiente a TP = 2, y el valor 0 para TP = 1. Para alzar el tercer bit de desplazamiento (atributo) basta con sumar 4 al valor de V (operación con la que en binario se pone a 1 el tercer bit). En la misma rutina 3000 se llaman después la 3500, que memoriza el desplazamiento (V) en la matriz B(100), y la 3400, que presenta el desplazamiento.

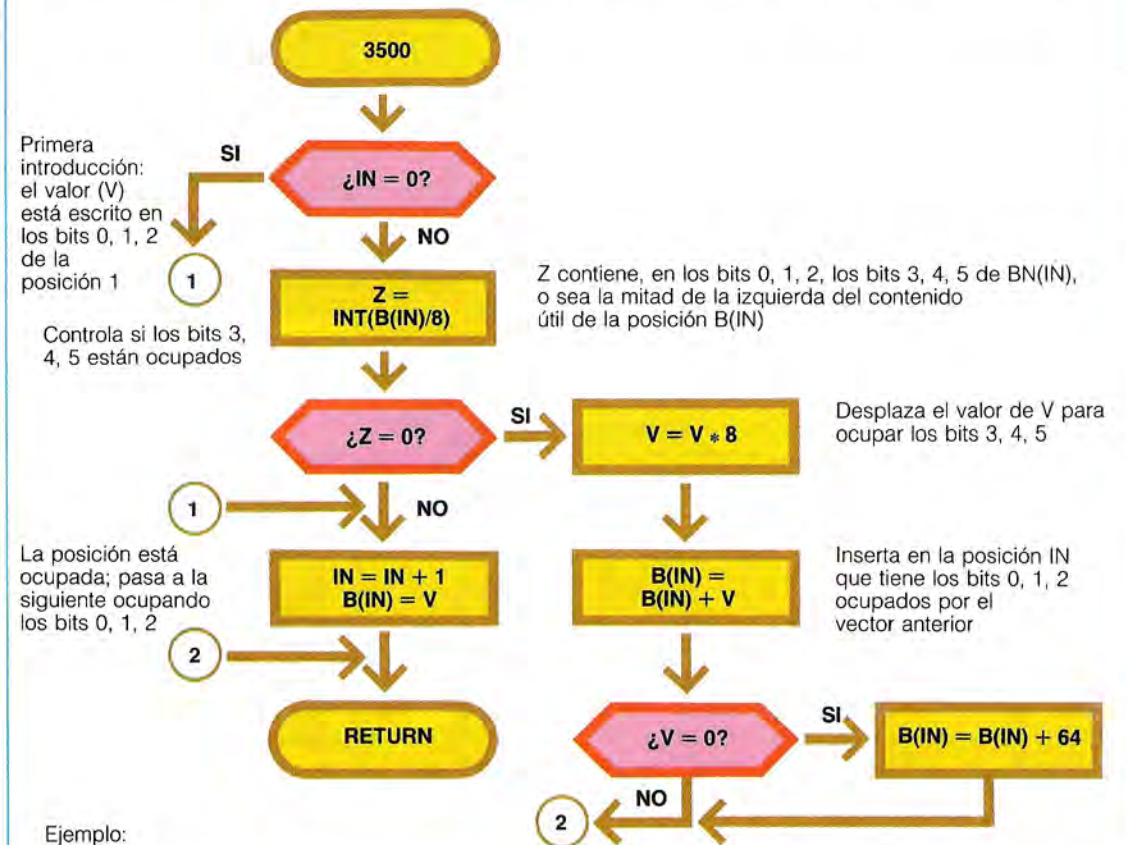
En la subrutina 3400 se activa el control sobre

los valores de X2 e Y2. Después de la salida de este control, si los valores superan la anchura de la pantalla de vídeo, no se tiene la presentación del desplazamiento, sino sólo su memorización (eventualmente con una señalización no indicada en el diagrama de flujo). Si se elige este camino en cuanto al programa, como se verá más adelante, permite variar la escala: si un desplazamiento sale de los límites de la pantalla, basta con activar el estado comandos y variar la escala de representación para eliminar el inconveniente.

La subrutina 3500, por el contrario, presenta algunas dificultades debidas al necesario «empaquetamiento» de los valores que va asumiendo la variable V. Abajo se muestra el diagrama de flujo. El índice N contiene la última posición ocupada en la matriz B(100). Inicialmente se ha puesto igual a cero y, por tanto, al primer desplazamiento, la subrutina salta las instrucciones intermedias y memoriza en N + 1, o sea en B(1), en la parte derecha [poniendo B(1) = V, el valor V ocupa las posiciones que le competen, o sea

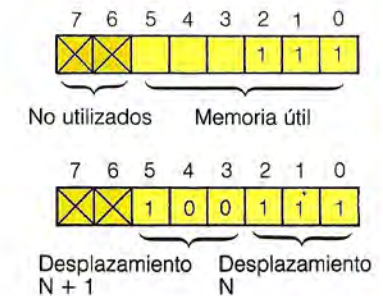
los bits de 0 a 2]. A la segunda llamada, IN ya no es cero (vale 1), y se activa la parte central. En estas instrucciones se analiza el contenido de B(IN) dividido por 8. Si el dato contenido en B(IN) sólo ocupa los bits 0, 1 y 2, el valor entero de la división es cero, de otro modo, no lo es. Si el resultado del cociente es cero, significa que la parte derecha (bits de 0 a 2) ya está ocupada (la memoria no puede contener todos ceros, puesto que IN es incremento antes de escribir); en este caso es necesario memorizar V en

SUBROUTINA DE MEMORIZACION EN FORMATO TABLA DE LAS FIGURAS



V = 7 (desplazamiento a la derecha en visión)
La memoria está vacía. La representación binaria de 7 es 111, y se escribe en los bits 0, 1, 2

V = 4 (desplazamiento hacia arriba en visión)
La memoria útil está ocupada en la parte derecha (INT(B(·)/8) = INT(7/8) = 0) y por tanto V debe trasladarse 3 bits: V = V * 8 = 32 = 1 0 0 0 0 0 (32 = 2^5). El contenido de la memoria se hace el representado.

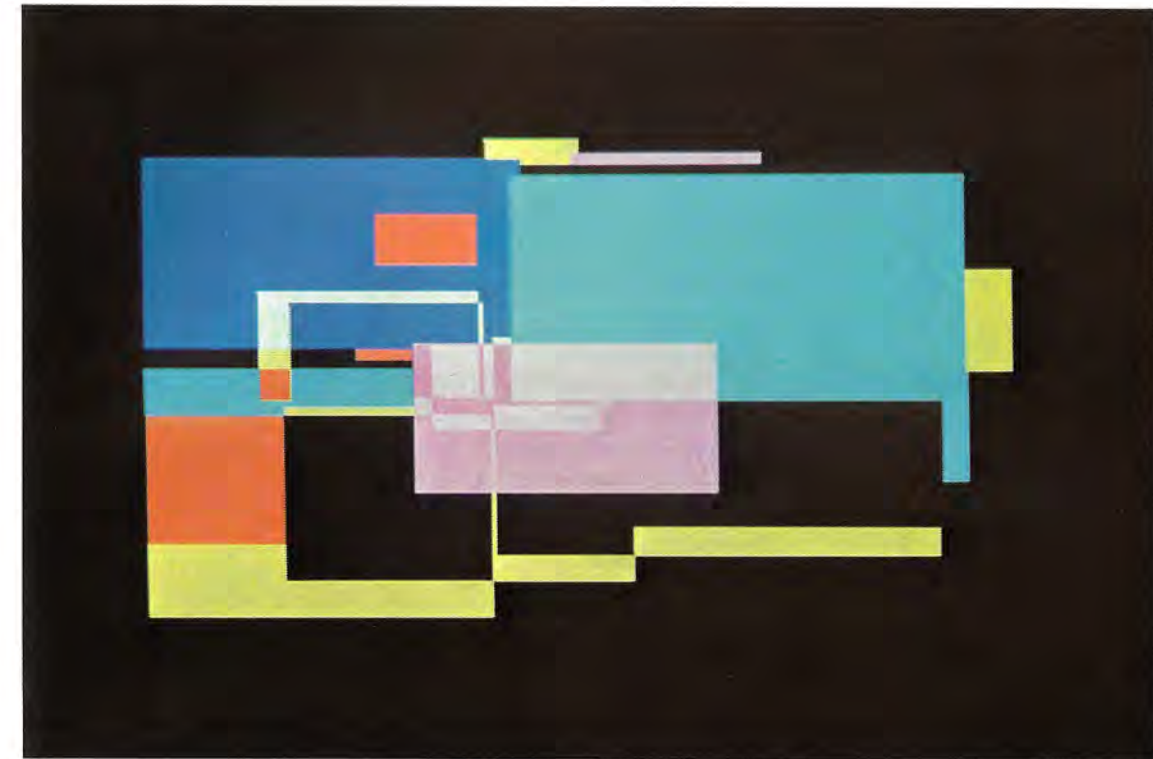


los bits 3, 4 y 5, y esto se obtiene multiplicando por 8 su valor y sumando el resultado al contenido anterior.

Abajo se ha representado el diagrama de flujo de la subrutina 4000 (gestión de las órdenes) que sólo contiene una serie de llamadas a otras subrutinas, cada una dedicada a uno de las posibles nueve órdenes. Unas eventuales implantaciones de nuevas funciones pueden obtenerse insertando otras instrucciones en la llamada direccionada (ON TP...), y naturalmente las subrutinas necesarias para su desarrollo. Las funciones de desplazamiento en visión o no en visión sólo necesitan la asignación del valor 0 o 1 al flag HC, y por tanto se realizan en la subrutina principal.

Obsérvese el método de la gran descomposición de las tareas. La subrutina 4000 habría podido contener todas las instrucciones de las otras, eliminando así la necesidad de implantar flags u otros indicadores para pasar a las subrutinas que siguen, pero sería muy compleja y difícilmente modificable. Además, utilizando subrutinas dedicadas a una función bien delimitada, pueden ahorrarse instrucciones reutilizando la misma subrutina en más puntos del programa; por ejemplo, la 4100, que es parte de la 4000, utiliza la 2000, y a su vez es reutilizada en la fase de reconstrucción del dibujo, después de la carga de la tabla de los desplazamientos desde el disco.

A continuación ilustraremos las lógicas utiliza-



Composición gráfica obtenida con simples instrucciones de alto nivel.

das en el interior de las principales subrutinas, mientras que el listado del programa completo se ha representado en las páginas 1609 y 1610.

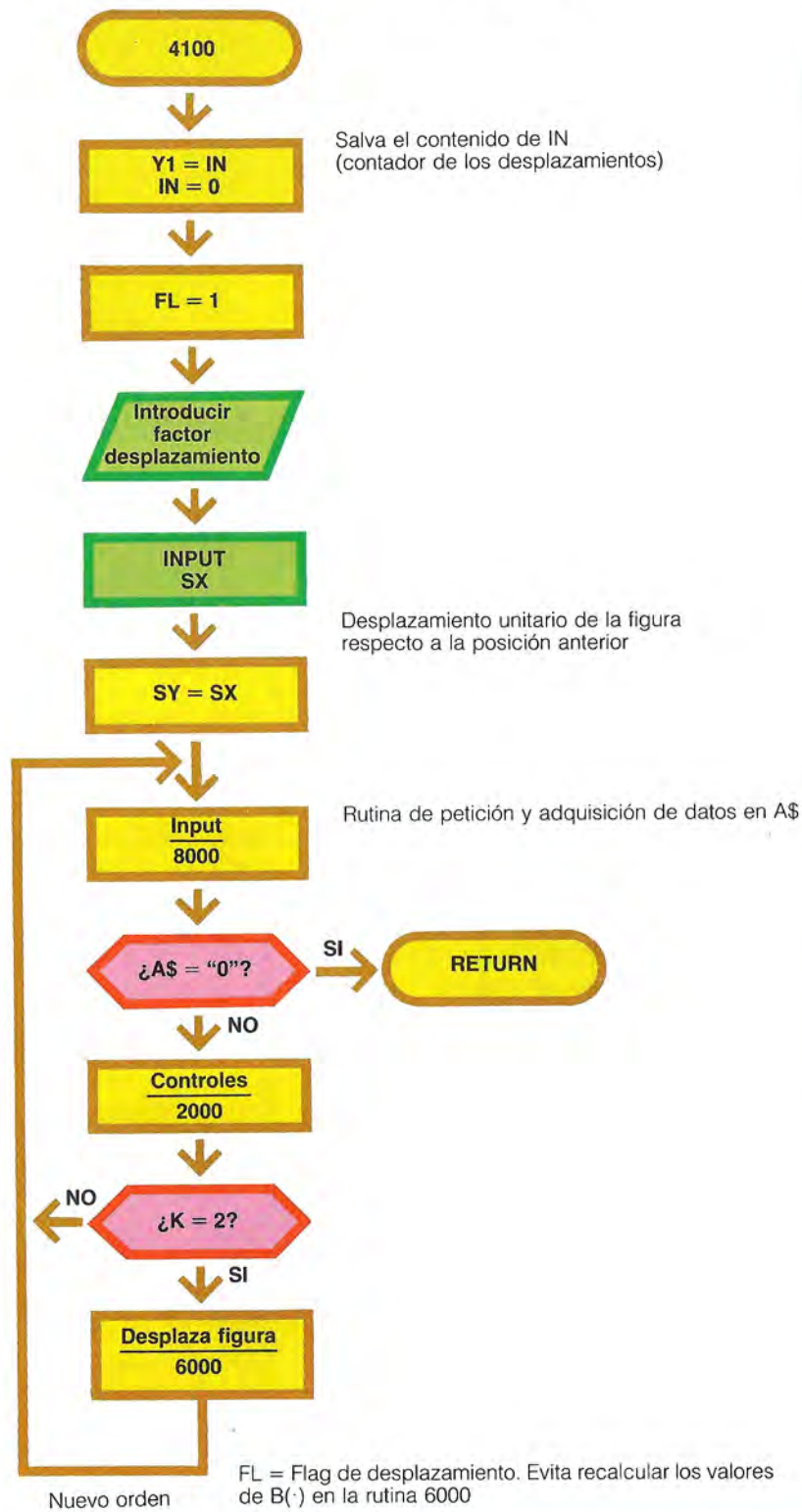
Gestión de los desplazamientos de conjunto (subrutina 4100). La función de estas subrutinas es la de desplazar todas las figuras a una parte cualquiera de la pantalla (no confundirla con la 3000, que se dedica a la gestión de los desplazamientos en la fase de la generación del dibujo). En la subrutina se ha previsto la salvaguarda de la tabla de los desplazamientos B(100) antes de la traslación del dibujo. Esta función puede evitarse, puesto que la modificación de la posición no implica variaciones en la tabla de los desplazamientos; sin embargo, se ha incluido para mostrar una metodología. Utilizando una matriz de apoyo, el original permanece sin alteración, y siempre es posible recuperarlo si los resultados no son satisfactorios. La primera función a realizar es la lectura del tipo de desplazamiento deseado. Se trata de una parte del programa similar a la utilizada para la generación de la tabla, con la única variante del reconocimiento del valor 0, que indica el final de la traslación. Las teclas que activan los

desplazamientos del dibujo son las mismas que las utilizadas para generar las tablas de los desplazamientos de la figura y, por tanto, son controladas con la misma subrutina 2000.

En este caso, sin embargo, no hay previstas órdenes; en consecuencia, si a la salida de la 2000 el valor de K es diferente de 2, se ignora (K = 2 significa que se ha pulsado una de las teclas previstas para los desplazamientos). Si la tecla se reconoce (y por tanto si es una de las cuatro previstas) se activa la subrutina 6000, que desplaza la figura. Dentro de la 6000, el programa se pone en espera de un nuevo desplazamiento o del valor 0.

Desplazamiento de la figura (subrutina 6000). Esta subrutina lee los desplazamientos que constituyen la figura [en B(100)] y la dibuja a partir de un punto cuyas coordenadas se han variado, respecto al origen de omisión X0, Y0, en una cantidad SX o SY, según el tipo de traslación deseado. La función se obtiene extrayendo de la tabla de los desplazamientos el atributo (en visión o no en visión) y el tipo de desplazamiento (0, 1, 2, 3) y utilizando las mismas subrutinas previstas en

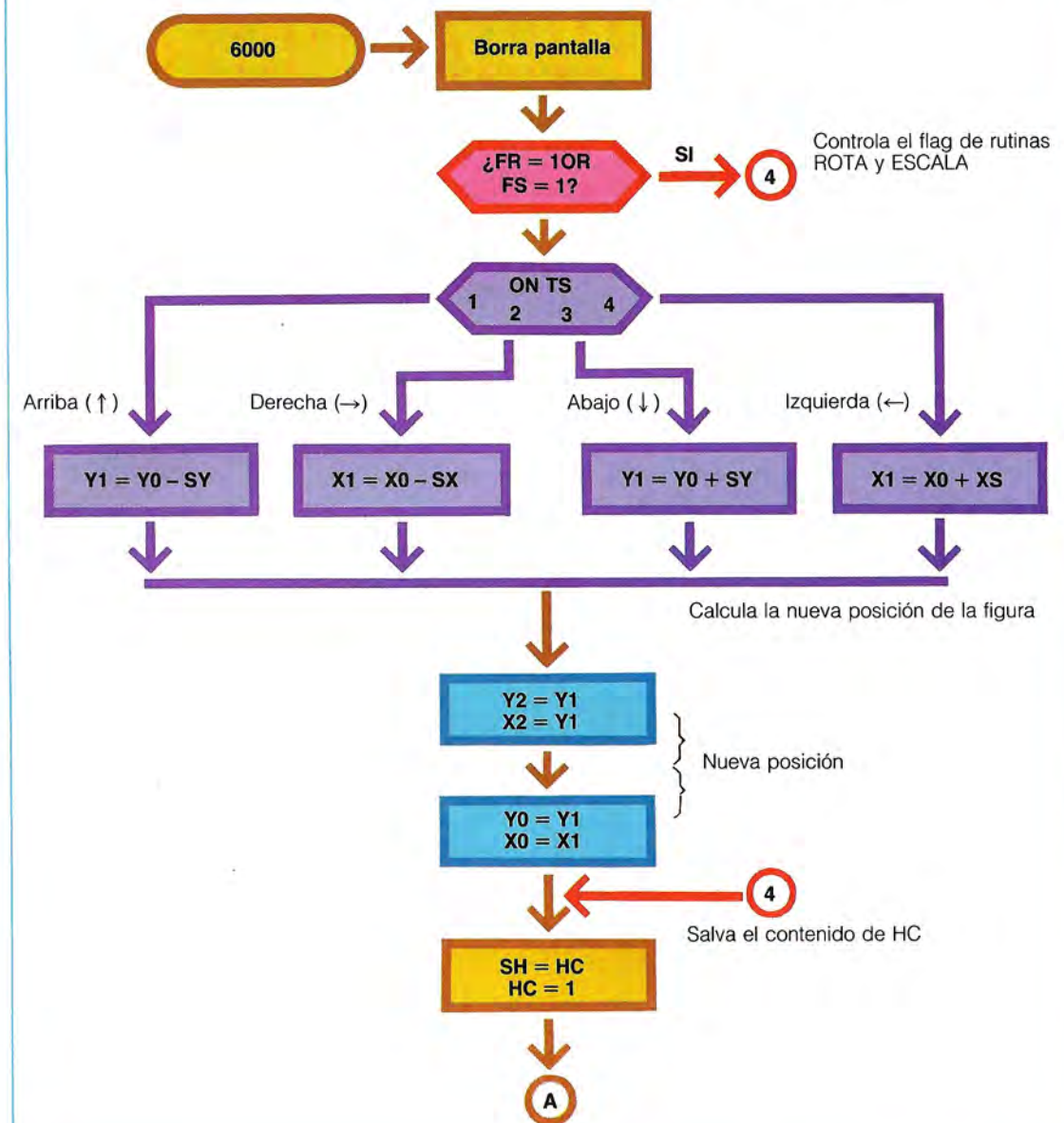
GESTION DE LOS DESPLAZAMIENTOS DE CONJUNTO

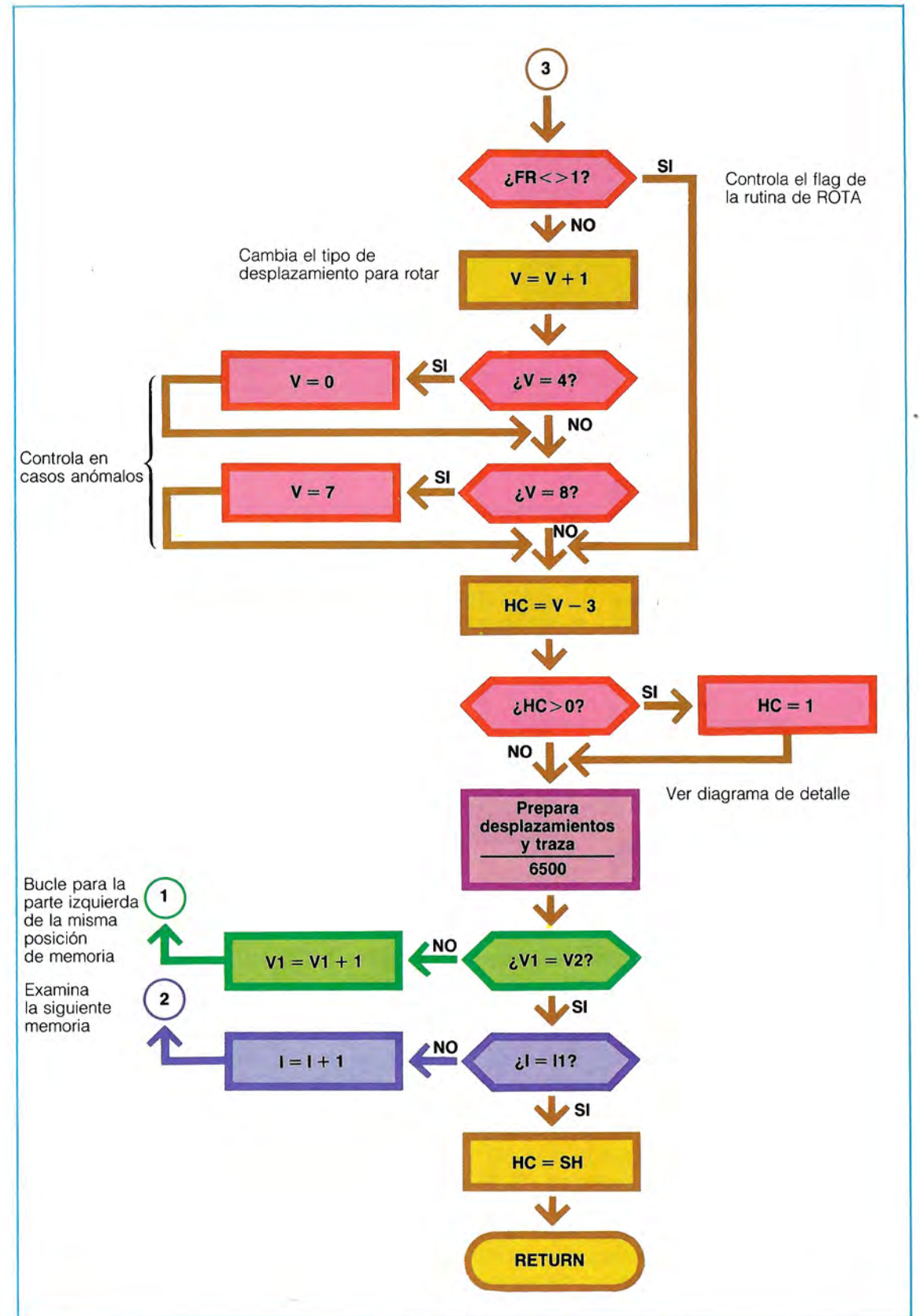
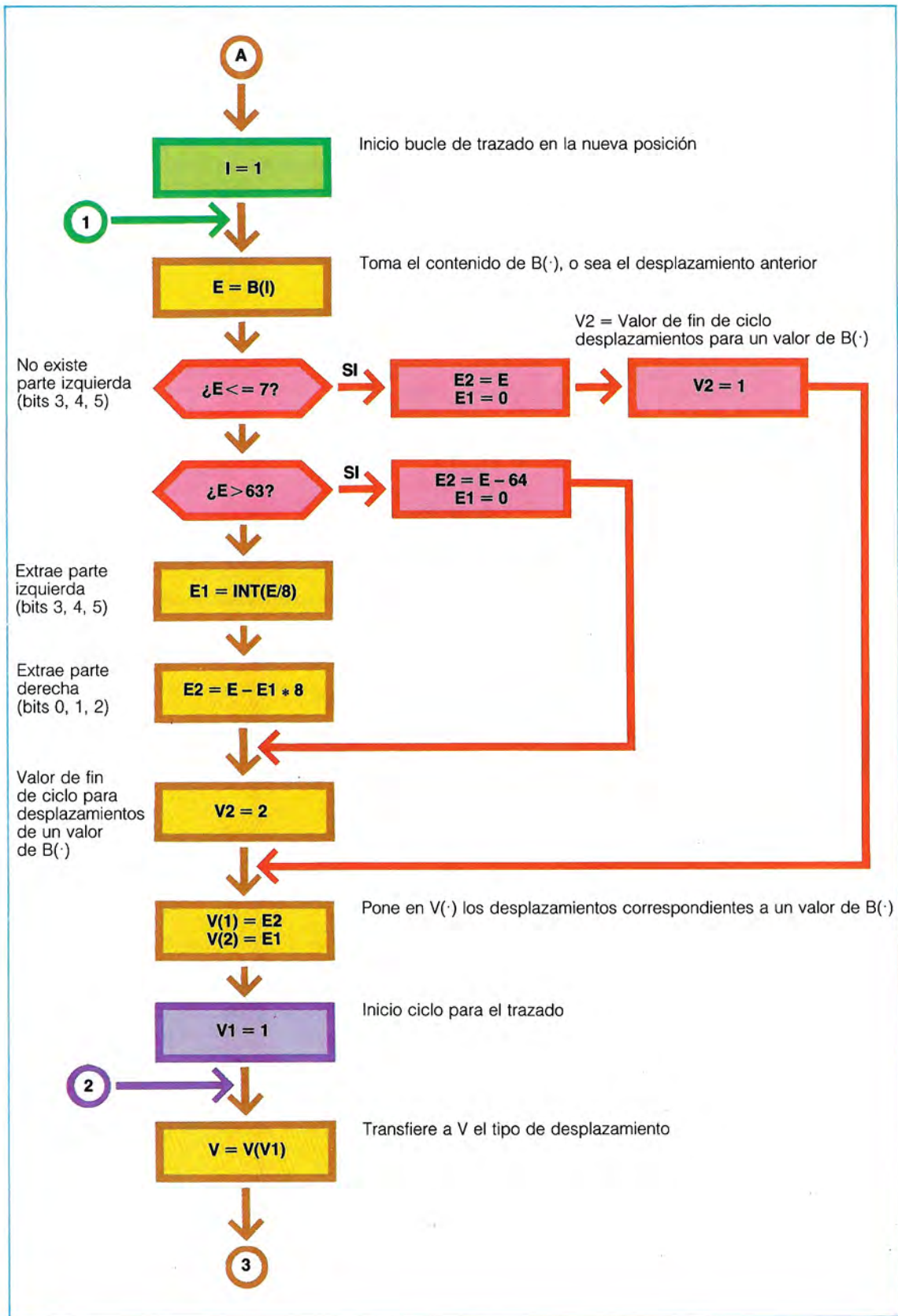


la fase de generación de la tabla. Básicamente se trata de una introducción simulada. Para no hacer prolijos el diagrama de flujo y el listado, se ha omitido el borrado del desplazamiento anterior. Activando la traslación de una figura se obtiene su repetición en un punto diferente. Por ejemplo, si la posición final se obtiene con 10 pasos intermedios, se tendrán otras tantas repeticiones de la figura, cada una desplazada respecto a la anterior en 1/10 del desplazamiento

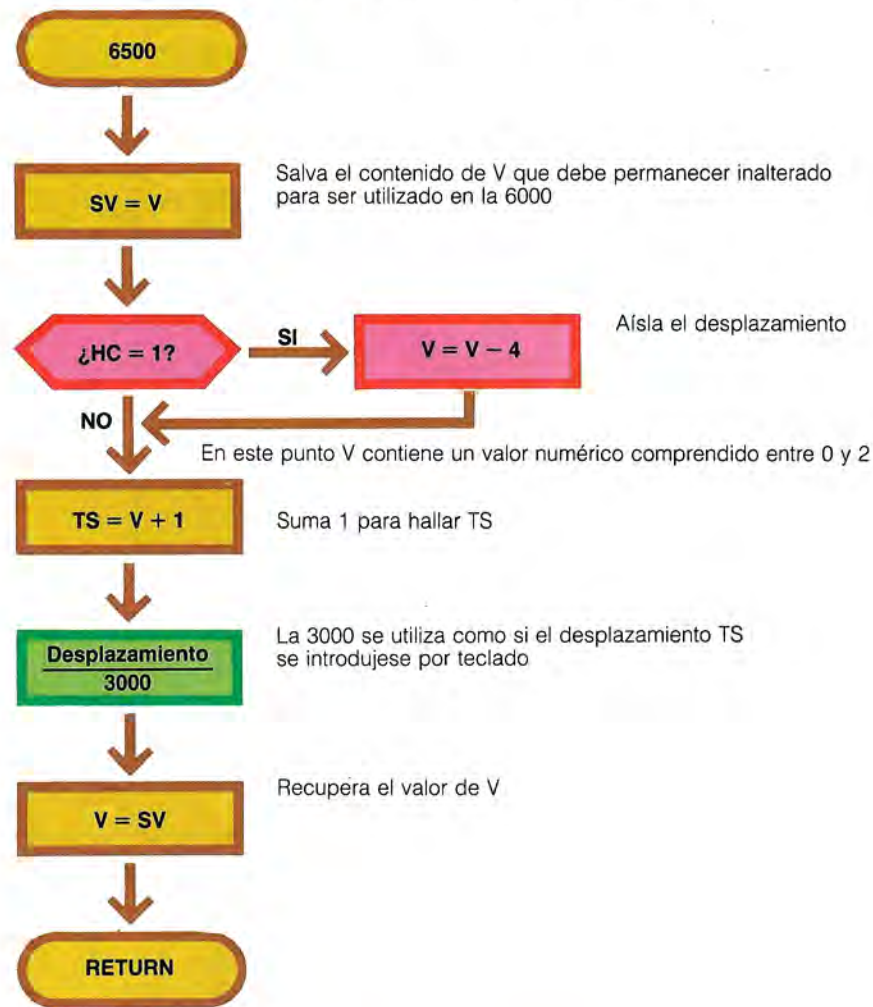
final. Para eliminar este inconveniente pueden utilizarse dos métodos. El primero consiste en borrar toda la página gráfica antes de activar el desplazamiento; sin embargo, así también se produce el borrado de la figura de partida. La modificación a aportar requiere la inserción de la sola instrucción de borrado de la memoria de vídeo antes de llamar la 6000. La reconstrucción de la figura en la posición original puede activarse al final de la 6000, cuando ya no

SUBROUTINA DE DESPLAZAMIENTO DE UNA FIGURA





PREPARACION DEL DESPLAZAMIENTO Y LLAMADA PARA LA PRESENTACION



es necesario realizar más borrados de la pantalla (ver los gráficos representados en la página anterior y arriba).

El segundo método, mucho más complicado, consiste en repasar los contornos de la figura anterior con un color igual al del fondo, obteniéndose así el borrado.

Además, existe una tercera posibilidad, pero que sólo es aplicable en algunas máquinas (por ejemplo en los sistemas Siprel y Apple). Si el ordenador dispone de dos páginas gráficas, puede utilizarse la primera para mantener la figura en la posición original y la segunda, siempre borrando cada vez, para las posiciones intermedias. Al final de la traslación se unen las

dos figuras (en posición original y trasladada) en una única página gráfica.

Rotación de la figura (subrutina 4500). Esta función se adapta poco a la estructura para desplazamientos en las cuatro direcciones dadas al programa, que no permite trazar líneas inclinadas. Sólo a título de ejemplo, en el gráfico de la página siguiente se ha indicado la lógica de rotación de 90° en sentido horario.

La rotación se obtiene simplemente sumando 1 a cada vector de desplazamiento, a condición de poner 0 si el resultado de esta suma es 4. En el ejemplo, la rotación se ha fijado para 90° en el sentido horario. Análogamente, la posición

de principio de la figura girada se implanta después a la de omisión (X0, Y0) más una cantidad constante (10 unidades). Para un empleo generalizado debe incluirse la introducción de una elección del usuario.

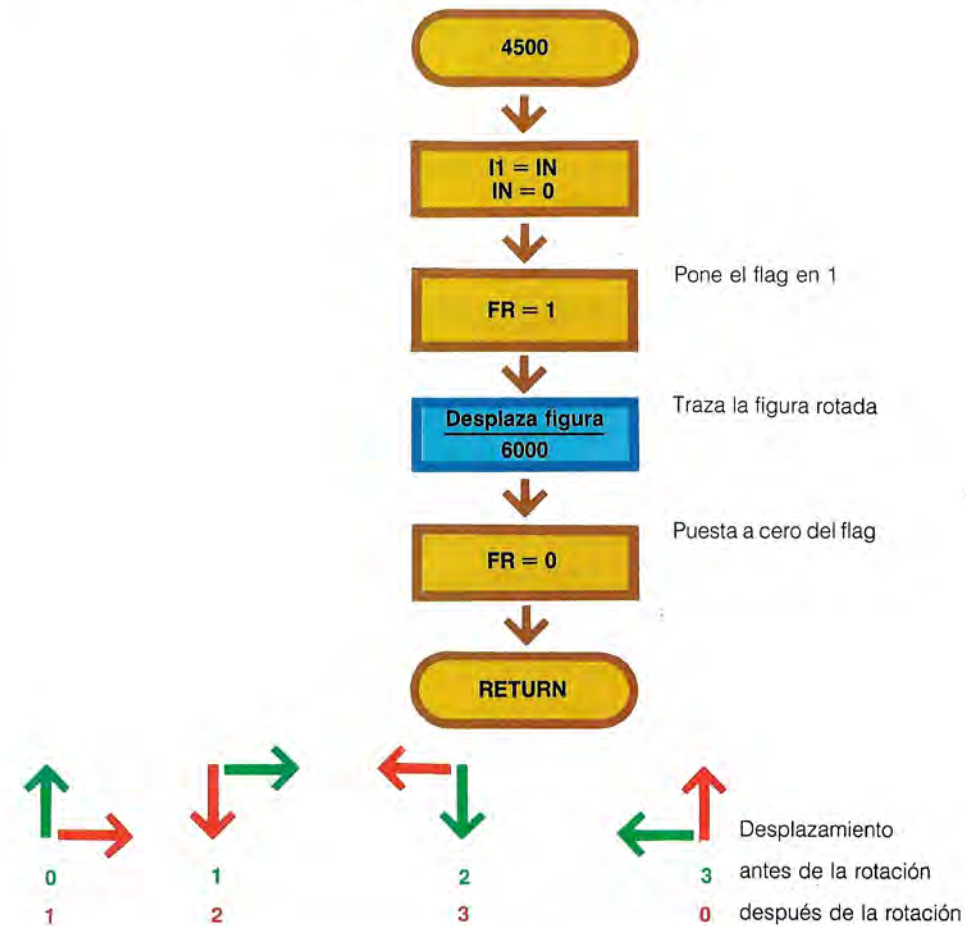
Transferencia al disco (subrutina 5000). La rutina sólo incluye la introducción del nombre del file que deberá contener B(100) y la escritura en disco. Por tanto, aquí sólo se indica el listado en las páginas 1612 a 1614.

Lectura del disco (subrutina 5200). Realiza funciones recíprocas a la 5000: una vez introducido el nombre del file que contiene los datos del dibujo a visualizar, transfiere el contenido a B(100) y llama la 6000 para la presentación. También de esta rutina sólo presentamos el listado (siempre en las páginas 1612 a 1614).



Una fase del funcionamiento del programa (ver listado en las págs. 1612 a 1614). Se ha seleccionado la página gráfica 1 que reserva cuatro líneas al texto. La ventana de texto se utiliza aquí para presentar el menú de los comandos (ver línea 118).

SUBROUTINA DE ROTACION



CREACION Y GESTION DE TABLAS DE LAS FIGURAS

```

1 REM =====
2 REM * CREACION *
3 REM * GESTION *
4 REM * FIGURAS EN *
5 REM * ALTA RESOL.*
6 REM =====
7 :
8 :
100 GOSUB 1000
110 :
111 REM =====
112 REM * LECTURA *
114 REM * CARACTERES*
114 REM =====
115 :
117 HGR : HOME
118 VTAB 21: PRINT "I=ARR/M=ABA'/
K=DER/J=IZQ/F=INV/V=VIS" PRINT
"S=DESPLAZA/R=ROTA/D=DISCO/L=
CARGA/C=BORRA"
120 GOSUB 8000
140 GOSUB 2000
150 ON K GOTO 160,170,190
160 GOTO 120
170 GOSUB 3000
180 GOTO 120
190 :
192 GOSUB 4000
200 GOTO 120
300 STOP
1000 :
1001 REM =====
1002 REM *INICIALIZA*
1003 REM =====
1004 :
1010 DIM B(100)
1020 IX = 10: IY = 10: Y0 = 50: X0 =
100: XM = 278: YM = 159: Y1 = Y
0: X1 = X0
1025 X2 = X1: Y2 = Y1: S = 1
1030 S(1) = 73: S(2) = 75: S(3) = 7
7: S(4) = 74
1035 0(1) = 70: 0(2) = 86: 0(3) = 8
3: 0(4) = 82: 0(5) = 68: 0(6) =
76
1040 0(7) = 67: 0(8) = 43: 0(9) = 4
5
1045 D$ = CHR$(4)
1100 RETURN
2000 :
2001 REM =====
2002 REM * CONTROL *
2003 REM * CARACTER *
2004 REM =====
2005 :
2010 K = 1: TS = 0: TP = 0: I = 1
2020 IF K < > 1 THEN RETURN
2030 IF S(I) = 0 THEN 2050
2040 IF ASC (A$) = S(I) THEN K =
2: TS = I: GOTO 2200
2050 IF 0(I) = 0 THEN 2070
2060 IF ASC (A$) = 0(I) THEN K =
3: TP = I: GOTO 2200
2070 IF I = 10 THEN RETURN
2080 I = I + 1: GOTO 2020
2100 :
2200 :
2210 GOTO 2070
3000 :
3001 REM =====
3002 REM * GESTION *
3003 REM * DESPLAZAMIENTOS*
3004 REM =====
3005 :
3010 ON TS GOTO 3020,3030,3040,3
050
3011 :
3012 REM <ARRIBA>
3013 :
3020 Y2 = Y1 - DY: V = 0: GOTO 310
0
3021 :
3022 REM <DERECHA>
3023 :
3030 X2 = X1 + DX: V = 1: GOTO 310
0
3031 :
3032 REM <ABAJO>
3033 :
3040 Y2 = Y1 + DY: V = 2: GOTO 310
0
3041 :
3042 REM <IZQUIERDA>
3043 :
3050 X2 = X1 - DX: V = 3
3060 :
3100 :
3105 IF HC = 1 THEN V = V + 4
3110 GOSUB 3400
3115 IF FL = 1 THEN RETURN
3150 GOSUB 3500
3160 RETURN
3400 :
3401 REM =====
3402 REM * TRAZA *
3403 REM *DESPLAZAMIENTO*
3404 REM =====
3405 :
3420 IF HC = 1 THEN HCOLOR= 3: GOTO
3440
3430 HCOLOR= 0
3440 IF Y2 > YM OR X2 > XM THEN
RETURN
3450 IF Y2 <= 0 OR X2 <= 0 THEN
RETURN
3460 HPLLOT X1,Y1 TO X2,Y2
3470 X1 = X2: Y1 = Y2
3480 RETURN
3500 :
3501 REM =====
3502 REM * MEMORIZA *
3503 REM * TABLAS *
3504 REM =====
3505 :
3510 IF IN = 0 THEN 3540
3520 Z = INT (B(IN) / 8)
3530 IF Z < > 0 THEN 3540
3535 V = V * 8: B(IN) = B(IN) + V:
IF V = 0 THEN B(IN) = B(IN)
+ 64: GOTO 3550
3537 GOTO 3550
3540 IN = IN + 1: B(IN) = V
3550 RETURN
4000 :
4001 REM =====
4002 REM * GESTION *
4003 REM * ORDENES *
4004 REM =====
4005 :
4010 IF TP < 1 OR TP > 9 THEN STOP

```

```

4011 :
4020 : ON TP GOTO 4030,4040,4050,4
060,4070,4080,4085,4090,4095
4030 HC = 0: RETURN
4040 HC = 1: RETURN
4050 GOSUB 4100: RETURN
4060 GOSUB 4500: RETURN
4070 GOSUB 5000: RETURN
4080 GOSUB 5200: RETURN
4085 RUN
4090 S = 3: GOSUB 9000: RETURN
4095 S = - 3: GOSUB 9000: RETURN
4100 :
4101 REM =====
4102 REM * GESTION *
4103 REM *DESPLAZAMIENTOS*
4104 REM =====
4105 :
4106 FL = 1
4140 I1 = IN
4145 VTAB 23: HTAB 12: INPUT "FAC
TOR DESPLAZAMIENTO? "; SX
4147 SY = SX
4148 VTAB 23: HTAB 22: PRINT "
"
4150 GOSUB 8000
4160 IF A$ = "0" THEN 4300
4170 GOSUB 2000
4180 IF K < > 2 THEN 4150
4190 GOSUB 6000
4200 GOTO 4150
4300 :
4310 :
4320 :
4325 IN = I1
4327 FL = 0
4330 RETURN
4500 :
4501 REM =====
4502 REM * ROTACION *
4503 REM =====
4504 :
4510 I1 = IN: IN = 0
4520 FR = 1: GOSUB 6000
4530 FR = 0
4540 RETURN
5000 REM =====
5001 REM *AL DISCO*
5002 REM =====
5003 :
5010 VTAB 23: HTAB 12: PRINT "NO
MBRE :
5020 VTAB 23: HTAB 19: INPUT "";
NM$
5030 VTAB 1: PRINT D$ "OPEN "NM$
5040 PRINT D$ "WRITE "NM$
5045 PRINT IN
5050 FOR I = 1 TO IN: PRINT B(I)
: NEXT I
5060 PRINT D$ "CLOSE "NM$: PRINT
D$
5070 RETURN
5200 REM =====
5201 REM *CARGA*
5202 REM =====
5203 :
5210 VTAB 23: HTAB 12: PRINT "NO
MBRE :
5220 VTAB 23: HTAB 19: INPUT "";
NM$
5230 VTAB 1: PRINT D$ "OPEN "NM$
5240 PRINT D$ "READ "NM$
5250 INPUT I1
5255 IN = I1
5260 FOR I = 1 TO I1: INPUT B(I)
: NEXT I
5270 PRINT D$ "CLOSE "NM$: PRINT
D$
5273 X0 = 100: Y0 = 50: X1 = X0: Y1 =
Y0: X2 = X1: Y2 = Y1: FL = 1
5275 GOSUB 6105: FL = 0
5280 RETURN
6000 :
6001 REM =====
6002 REM *DESPLAZAMIENTO*
6003 REM * FIGURA *
6004 REM =====
6005 :
6007 HGR
6008 IF FR = 1 OR FS = 1 THEN 61
05
6010 ON TS GOTO 6020,6030,6040,6
050
6020 Y1 = Y0 - SY: GOTO 6100
6030 X1 = X0 + SX: GOTO 6100
6040 Y1 = Y0 + SY: GOTO 6100
6050 X1 = X0 - SX
6100 X2 = X1: Y2 = Y1: Y0 = Y1: X0 =
X1
6105 SH = HC: HC = 1
6110 FOR I = 1 TO I1
6120 E = B(I)
6130 IF E <= 7 THEN E1 = 0: E2 =
E: V2 = 1: GOTO 6155
6135 IF E > 63 THEN E2 = E - 64:
E1 = 0: GOTO 6153
6140 E1 = INT (E / 8)
6150 F2 = E - E1 * 8
6153 V2 = 2
6155 V(1) - E2: V(2) = E1
6160 V1 = 1
6165 V = V(V1)
6167 IF FR < > 1 THEN 6175
6169 V = V + 1
6170 IF V = 4 THEN V = 0
6172 IF V = 8 THEN V = 4
6175 HC = V - 3: IF HC > 0 THEN H
C = 1
6180 GOSUB 6500
6185 VTAB 23: HTAB 35: PRINT TS
6190 :
6200 IF V1 < > V2 THEN V1 = V1 +
1: GOTO 6165
6210 NEXT I
6220 IF I < I1 THEN I = I1: NEXT
I
6230 HC = SH: RETURN
6500 :
6501 REM =====
6502 REM * PREPARACION *
6503 REM * DESPLAZAMIENTO*
6504 REM =====
6505 :
6510 SV = V: IF HC = 1 THEN V = V
- 4
6520 TS = V + 1: GOSUB 3000
6530 V = SV: RETURN
8000 :
8001 REM =====
8002 REM * INPUT *

```

```

8003 REM =====
8004 :
8010 VTAB 23: HTAB 12: PRINT "IN
      SERTAR <?> COMANDO"
8020 VTAB 23: HTAB 22: GET A$
8021 VTAB 23: HTAB 35: INVERSE
      PRINT A$: NORMAL
8022 VTAB 23: HTAB 1: PRINT "
      "
8024 VTAB 23: PRINT "X="X1"Y="Y1
8025 IF ASC (A$) = 13 THEN TEXT

      : HOME : END
8030 RETURN
9000 REM =====
9001 REM * ESCALA *
9002 REM =====
9003 :
9010 FS = 1
9015 DY = DY + S
9020 DX = DX + S: I1 = IN: IN = 0
9030 GOSUB 6000
9040 FS = 0: RETURN

```

Programas de utilidad para la creación de figuras gráficas

El ejemplo considerado de la construcción de una tabla de los desplazamientos ilustra un método que permite construir una imagen gráfica en los ordenadores que no tienen instrucciones expresamente dedicadas a este fin.

Los microordenadores que prevén las tablas de las figuras utilizan dos tipos fundamentales de instrucciones. El primer tipo pide los desplazamientos elementales con las simbologías y las convenciones vistas en el ejemplo; en cambio, el segundo utiliza una cadena de comandos en la que hay especificados, uno después de otro, los desplazamientos simples.

El primer tipo está en los ordenadores basados en la CPU 6502, que pertenecen a la familia Apple y compatibles; en cambio, el segundo es común a varias máquinas.

Para generar una tabla de las figuras con el primer tipo de máquina, el mejor modo es escribir directamente en la memoria los vectores de los desplazamientos con la instrucción POKE. Por tanto, la metodología prevé dibujar antes la tabla sobre una hoja de papel, tomando los desplazamientos elementales que la constituyen, y después cargándola en memoria según el formato visto.

Utilizando el software también es posible simular, en los ordenadores que no las prevén, las instrucciones de alto nivel dedicadas a la gestión y a la presentación de figuras complejas. En estas páginas presentamos dos ejemplos de notable utilidad práctica.

El primer programa, que hemos llamado «Shape Editor» siguiendo una terminología muy difundida, permitirá construir directamente en la pantalla figuras (shape) también muy complejas y memorizar con un nombre convencional en el

disco la tabla de los desplazamientos correspondientes. El segundo programa, que llamaremos «Shape Loader», permitirá en cambio cargar del disco y presentar una figura construida anteriormente.

Shape Editor. El diagrama de flujo del Shape Editor puede verse en la figura de la página siguiente. Prevé la posibilidad de componer en la pantalla un dibujo utilizando un conjunto de teclas que activan el desplazamiento, en visión o no, de la punta de escritura imaginaria.

El bloque para la adquisición y la ejecución de los comandos de desplazamiento se ha detallado en la figura de la página 1616.

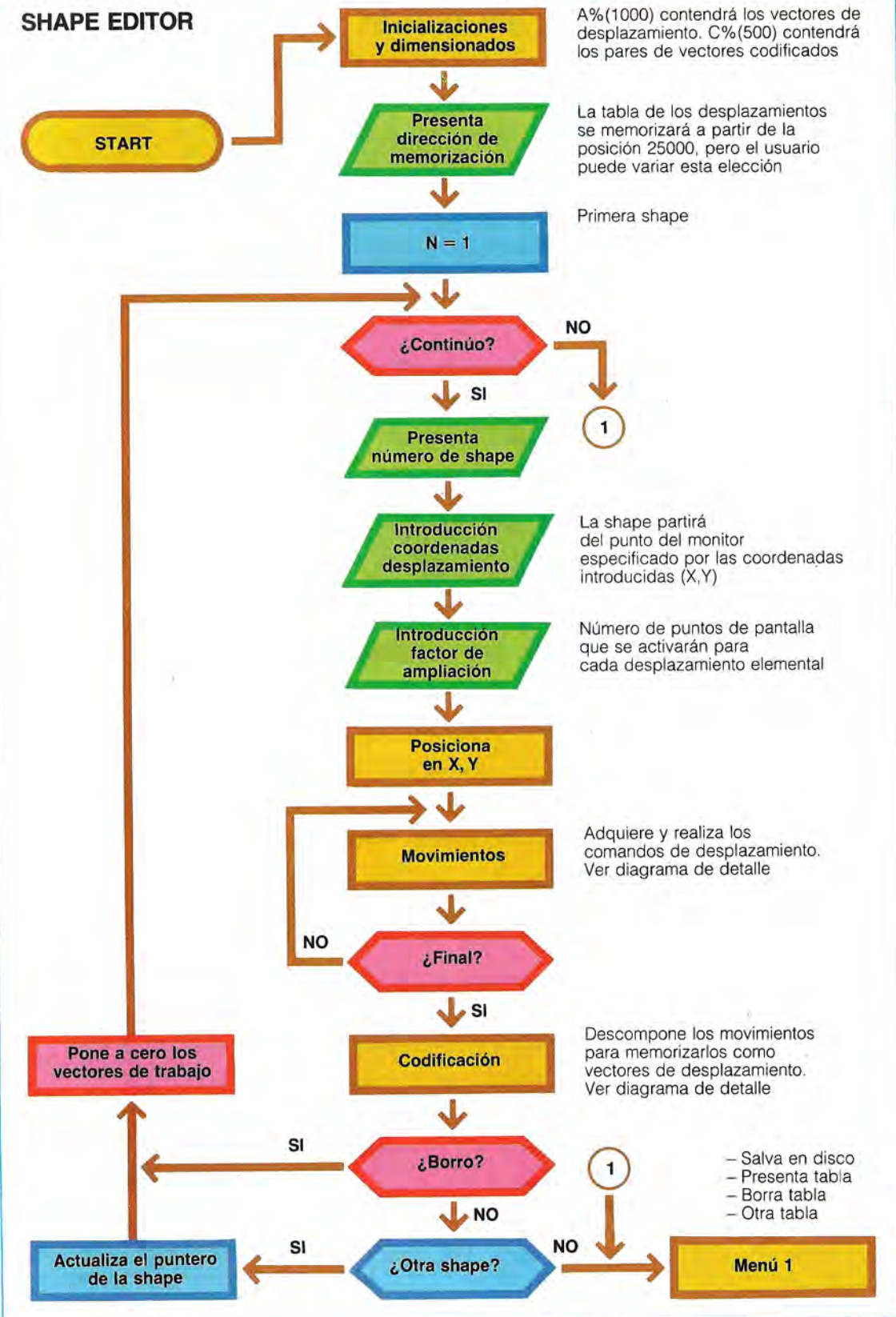
Terminado el trazado de una figura gráfica (shape) se entra en el bloque de codificación (figura de la página 1617), en el que el conjunto de los desplazamientos de la punta de escritura, memorizado en la matriz A% se ha convertido en el conjunto de vectores gráficos, que se memoriza en la matriz C%. Al final de la codificación, el conjunto de vectores de desplazamiento contenido en C% se carga en la memoria.

En este momento, el usuario podrá elegir la construcción de otra shape, que será tratada de la misma manera por el programa.

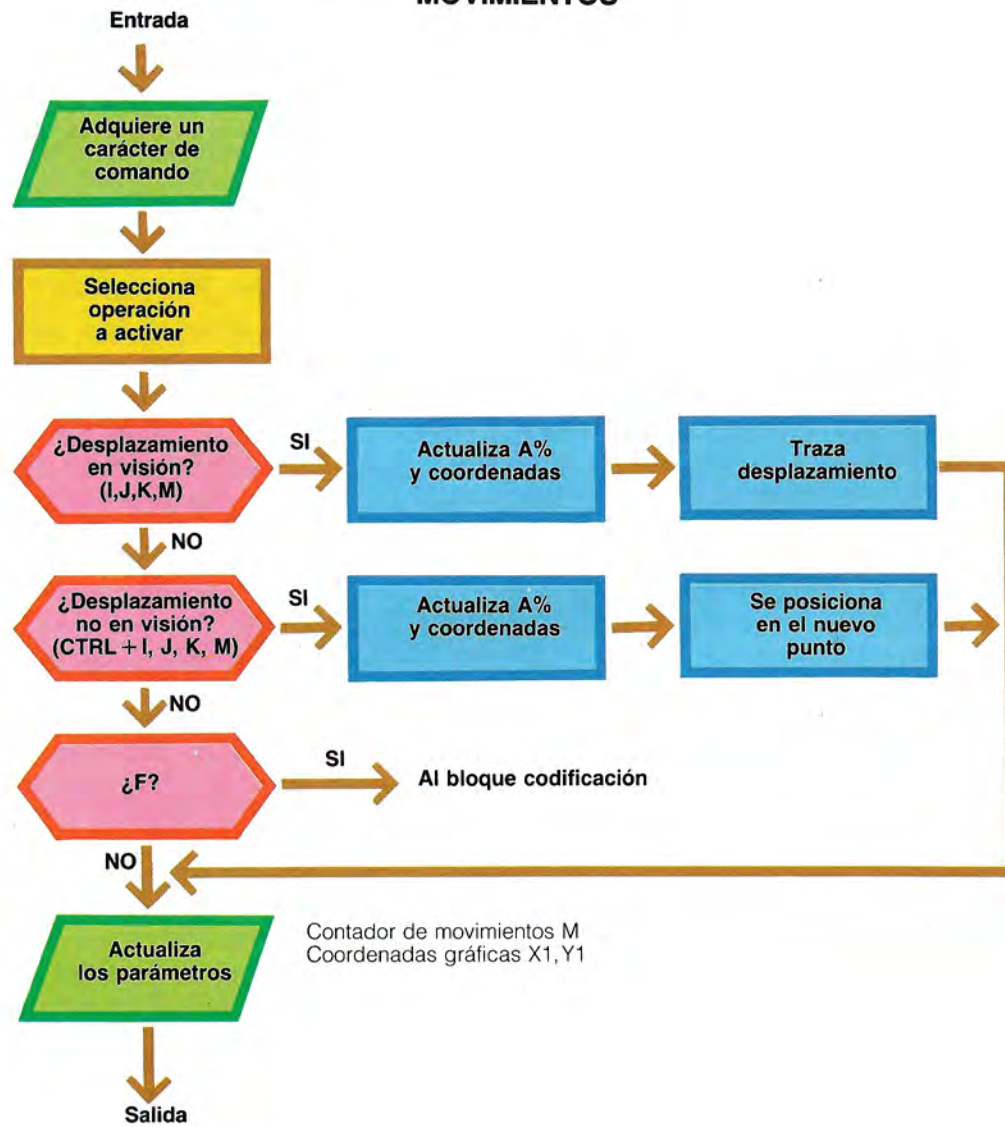
Después de haber terminado el trazado de las figuras que interesan, entrando en el menú se tiene la posibilidad de memorizarlas en disco bajo un nombre global, que designa precisamente la tabla de las shapes (shape table). En el ámbito de cada tabla, las diferentes shapes se distinguen por su número progresivo.

El listado del programa, en la versión para Sirel, Apple y compatibles, se ha representado en las páginas 1618 a 1621, mientras que la relación de las variables utilizadas se ha representado en la tabla de la página 1623.

SHAPE EDITOR



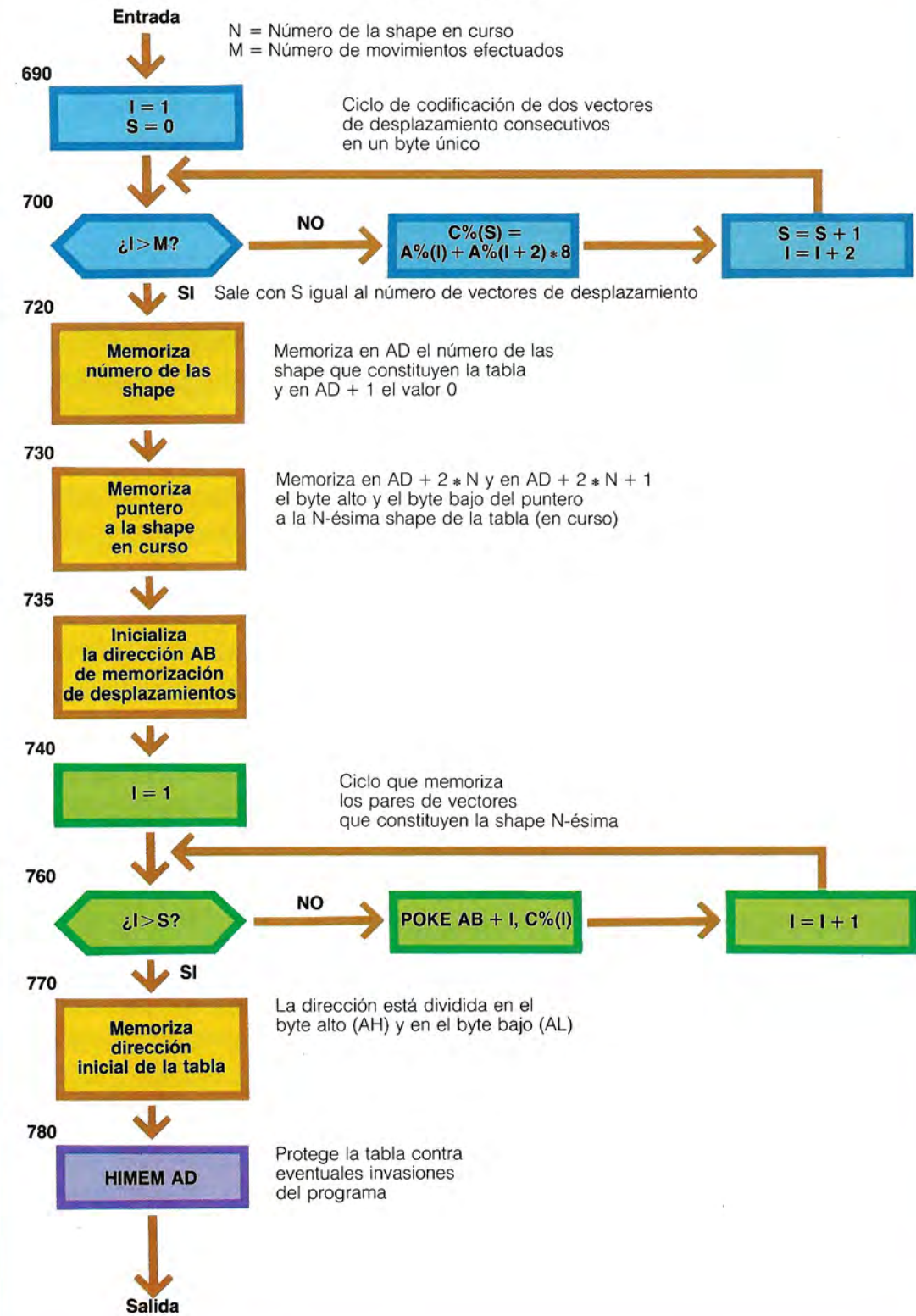
MOVIMIENTOS



CARACTERES DE CONTROL DE LOS DESPLAZAMIENTOS

Carácter	Modalidad	Dirección	Código en A%
CTRL + I	No en visión	Hacia arriba	0
CTRL + K	No en visión	A la derecha	1
CTRL + M	No en visión	Hacia abajo	2
CTRL + J	No en visión	A la izquierda	3
I	En visión	Hacia arriba	4
K	En visión	A la derecha	5
M	En visión	Hacia abajo	6
J	En visión	A la izquierda	7

CODIFICACION



SHAPE EDITOR

Versión Siprel, Apple y compatibles

```

70  REM PUEDE CONTENER HASTA
80  REM 30 SHAPES.
90
:
100 DIM AX(1000),CY(500)
110 TEXT
: HOME
120 INVERSE
: PRINT "SHAPE EDITOR"
: NORMAL
140 PRINT
: PRINT
150 AD = 25000
160 PRINT "DIRECCION DE LA SHAPE TABL
      E = 25000 "
: PRINT "QUIERE CAMBIAR (S/N) ":
: INPUT Z$
170 IF LEFT$(Z$,1) = "S" THEN
      PRINT
: INPUT "NUEVO VALOR";AD
180 PRINT
: PRINT "OK! ";AD
190 AH = INT (AD / 256)
: AL = AD - AH * 256
200 BL = 62
: BH = 0
: BB = 62
210 N = 1
220 HOME
230 PRINT
: PRINT " SHAPE NUMERO ";
: INVERSE
: PRINT N
: NORMAL
240 PRINT
: PRINT "<1> CONTINUO"
250 PRINT
: PRINT "<2> STOP"
260 PRINT
270 PRINT
: INPUT "ELIJA: ";A$
280 IF VAL (A$) = 2 THEN N = N - 1
: BB = BB - S
: GOTO 1000
290 IF VAL (A$) < > 1 THEN 220
300 HGR
310 HOME
320 S = 1
330 M = 1
340 VTAB 22
: INPUT " COORDENADAS INICIALES? ";X,Y

350 HOME
: VTAB 22
: INPUT "FACTOR DE AMPLIACION (
      1-10) ? ";IN
360 IF IN < 1 OR IN > 10 THEN 350
370 HOME
: VTAB 21
380 PRINT "<I>,<J>,<K>,<M> PARA DIBU
      JAR (JUNTO A <CTRL> PARA DESPLA
      ZAR SIN DIBUJAR) <F> PARA TERMI
      NAR.;"
390
:
400 REM PUNTO INICIAL
410
:

```

```

420 HCOLOR= 3
430 HPLOT X,Y
440 X1 = X
: Y1 = Y
450
:
460 REM MOVIMIENTOS
470
:
480 HTAB 1
: VTAB 1
: GET A$
490 HTAB 1
: VTAB 24
: PRINT "MOVIMIENTOS:";
: INVERSE
: PRINT M;
: NORMAL
500 IF A$ = "I" THEN AX(M) = 4
: Y = Y - IN
: GOTO 600
510 IF A$ = "M" THEN AX(M) = 6
: Y = Y + IN
: GOTO 600
520 IF A$ = "J" THEN AX(M) = 7
: X = X - IN
: GOTO 600
530 IF A$ = "K" THEN AX(M) = 5
: X = X + IN
: GOTO 600
540 IF A$ = CHR$(9) THEN AX(M) = 0
: Y = Y - IN
: GOTO 610
550 IF A$ = CHR$(13) THEN AX(M) = 2
: Y = Y + IN
: GOTO 610
560 IF A$ = CHR$(10) THEN AX(M) = 3
: X = X - IN
: GOTO 610
570 IF A$ = CHR$(11) THEN AX(M) = 1
: X = X + IN
: GOTO 610
580 IF A$ = "F" THEN 670
590 GOTO 480
600 HPLOT X1,Y1 TO X,Y
: GOTO 630
610 IF IN = 1 THEN 630
620 HPLOT X,Y
630 M = M + 1
640 X1 = X
: Y1 = Y
650 GOTO 460
660

670 REM CODIFICA
680
690 FOR I = 1 TO M STEP 2
700 CX(S) = AX(I) + AX(I + 1) * 8
: S = S + 1
710 NEXT
720 POKE AD,N
: POKE AD + 1,0
730 POKE AD + 2 * N,BL
: POKE AD + 2 * N + 1,BH
735 AB = AD + BB - 1
740 FOR I = 1 TO S

```

```

750 POKE AB + I,C%(I)
760 NEXT
770 POKE 232,AL
: POKE 233,AH
780 HIMEM: AD
790
:
800 REM MENU 2
810
:
820 HOME
: VTAB 21
830 PRINT "SHAPE NUMERO ";N
: PRINT
840 PRINT "<1>BORRO <2>OTRA <3>
STOP"
850 PRINT
860 PRINT "ELIJA!":
: GET Z$
870 Z = VAL (Z$)
: IF Z < 1 OR Z > 3 THEN 800
880 ON Z GOTO 900,910,1000
890 GOTO 910
900 N = N - 1
: BB = BB - S
910 N = N + 1
: BB = BB + S
920 BH = INT (BB / 256)
: BL = BB - BH * 256
930 FOR I = 1 TO S
C%(I) = 0
: NEXT
940 FOR I = 1 TO M
: A%(I) = 0
: NEXT
950 TEXT
: HOME
: PRINT "PARA HACER LA SHAPE NUMERO ";
N
960 VTAB 5
: GOTO 230
970
:
980 REM MENU 1
990
:
1000 TEXT
: HOME
: VTAB 5
1010 F1$ = "E"
: IF N = 1 THEN F1$ = "A"
1020 PRINT
: PRINT "HAS DIBUJADO ";N;"FIGUR
A";F1$
1030 PRINT
: PRINT "DIRECCION DE LA TABLA !":
AD
1040 BT = BB + S
1050 IF N = 0 THEN BT = 0
1060 PRINT "LONGITUD ";BT
1070 PRINT
: PRINT
1080 PRINT "<1> SALVA LA TABLA EN DIS
CO"
1090 PRINT "<2> PRESENTA EN SECUENCIA
"
1100 PRINT "<3> BORRA TABLA"
1110 PRINT "<4> OTRA TABLA"
1120 PRINT "<5> FIN"
1130 PRINT
1140 PRINT "ELIJA !":
: GET Z$
1150 Z = VAL (Z$)

```

```

1160 IF Z < 1 OR Z > 5 THEN 1000
1170 ON Z GOTO 1210,1280,1490,1190,118
0
1180 HOME
: END
1190 CLEAR
: GOTO 100
1200
:
1210 REM SALVA EN DISCO
1220
:
1230 IF N = 0 THEN HOME
: VTAB 12
: HTAB 10
: FLASH
: PRINT "NO HAY FIGURAS"
: NORMAL
: GET Z$
: GOTO 1000
1240 PRINT
: INPUT "CON QUE NOMBRE? ";A$
1250 PRINT CHR$(4)"BSAVE",A$;"A";AD;"
,L";BB + S
1260 GOTO 1000
1270
:
1280 REM PRESENTA TABLA
1290
:
1300 FOR I = 1 TO N
1310 RO = 0
: SC = 1
1320 HGR
: HCOLOR= 3
: ROT= RO
: SCALE= SC
1330 HOME
1340 VTAB 22
: PRINT "SHAPE NUMERO ";I;" SCALE
= ";SC;" ROT = ";RO
1350 VTAB 23
: PRINT "PARA CAMBIAR: <S>SCALE, <R>
ROT <F>FIGURA (CON <CTRL> <TECL
A> SE VUELVE ATRAS)";
1360 DRAW I AT 140,80
1370 GET A$
1380 IF A$ = "S" THEN SC = SC + 1
: IF SC > 255 THEN SC = 255
1390 IF A$ = "R" THEN RO = RO + 1
: IF RO > 255 THEN RO = 255
1400 IF A$ = CHR$(18) THEN RO = RO
- 1
: IF RO < 0 THEN RO = 0
1410 IF A$ = CHR$(19) THEN SC = SC
- 1
: IF SC < 1 THEN SC = 1
1420 IF A$ = "F" THEN 1440
1430 GOTO 1320
1440 NEXT
1450 GOTO 1000
1460
:
1470 REM BORRA TABLA
1480
:
1490 PRINT
: PRINT "BORRO EFECTIVAMENTE <S/N> ? ";
: GET Z$.
1500 IF Z$ = "S" THEN CLEAR
: GOTO 100
1510 GOTO 1000

```

SHAPE EDITOR

He aquí algunas imágenes de vídeo tomadas durante el funcionamiento del programa Shape Editor.

El programa empieza implantando la dirección de memoria a partir de la cual se memorizará la tabla que se crea.
El usuario tiene también la posibilidad de modificar el valor presentado.



La tabla está inicialmente vacía, y el programa presenta el número de la shape en curso de proceso.



Después de la introducción de los necesarios parámetros (coordenadas iniciales, factor de ampliación), el programa pasa a la modalidad gráfica, presentando el menú de los comandos. Se ha dibujado el símbolo gráfico de un condensador.



La figura se ha completado y puede pasarse a otra. Cuando la tabla está completa puede transferirse al disco.



PRINCIPALES VARIABLES DEL PROGRAMA SHAPE EDITOR

A%	Matriz que contiene los desplazamientos codificados
C%	Matriz que contiene los pares de vectores gráficos codificados
AD	Dirección de partida (en memoria) de la shape table (parte baja = AL; parte alta = AH)
BB	Dirección de partida de la nueva shape (parte baja = BL; parte alta = BH)
IN	Factor de ampliación
M	Número de movimientos que componen la shape
N	Número de la shape en curso
RO	Flag de rotación para una shape en presentación
SC	Flag de escala para una shape en presentación
X,Y	Coordenadas gráficas
X1,Y1	

Shape Loader. La memorización en disco de una shape table tiene un sentido si después existe un programa que pueda utilizarla.

La definición de un cierto número de figuras gráficas agrupadas en una tabla sirve efectivamente para disponer de un menú de símbolos que pueden componerse de manera diferente para formar un dibujo también complejo.

Por tanto, las potencialidades del Shape Editor deben estar integradas proyectando un programa que pueda realizar las operaciones de lectura de una shape table y de composición en la pantalla de sus elementos. El diagrama de flujo de un programa que puede activar estas funciones se muestra en la figura de arriba de la página siguiente.

La presentación del menú inicial permite al usuario seleccionar la opción deseada entre las previstas; en respuesta, el programa transfiere el control a la sección interesada.

La subrutina 300, detallada en la figura de abajo de la página siguiente, carga en la memoria una tabla memorizada anteriormente en disco, y luego el programa vuelve a presentar el menú.

La subrutina 900 (figura de la página 1625) permite presentar una de las shapes contenidas en la tabla. También después de estas operaciones, el programa vuelve a presentar el menú. Por tanto, puede entrarse en la fase de construcción de una imagen compuesta introduciendo la opción 3 (subrutina 500). La subrutina 500 selecciona el modo gráfico y pide el número de la shape a presentar y el punto de pantalla en que debe realizarse la presentación. Una vez obtenidos estos parámetros presenta la imagen pedida y pasa el control a la rutina 580, colocada previamente a la adquisición y a la ejecución

de los comandos correspondientes a la composición de las shapes. Los comandos reconocidos por la 580 y sus significados se compendian en la tabla de la página 1626.

Además de las funciones escritas existe también la posibilidad de pedir el catálogo (índice o directorio) del disco y la de grabar la imagen compuesta en el disco.

El listado completo del programa puede verse en las páginas 1626 a 1630, mientras que en la tabla de la página 1630 se han relacionado las principales variables utilizadas.

Los dos programas presentados se han desarrollado en un ordenador Siprel, y sólo pueden funcionar en los ordenadores de la familia Apple y compatibles.

Instrucciones Basic para la gestión de figuras gráficas

Una vez memorizada, en algunas máquinas, una figura gráfica puede elaborarse como dato de conjunto mediante instrucciones de alto nivel, como por ejemplo

DRAW	Dibuja la figura
XDRAW	Borra
ROT	Rota
SCALE	Varía el factor de escala

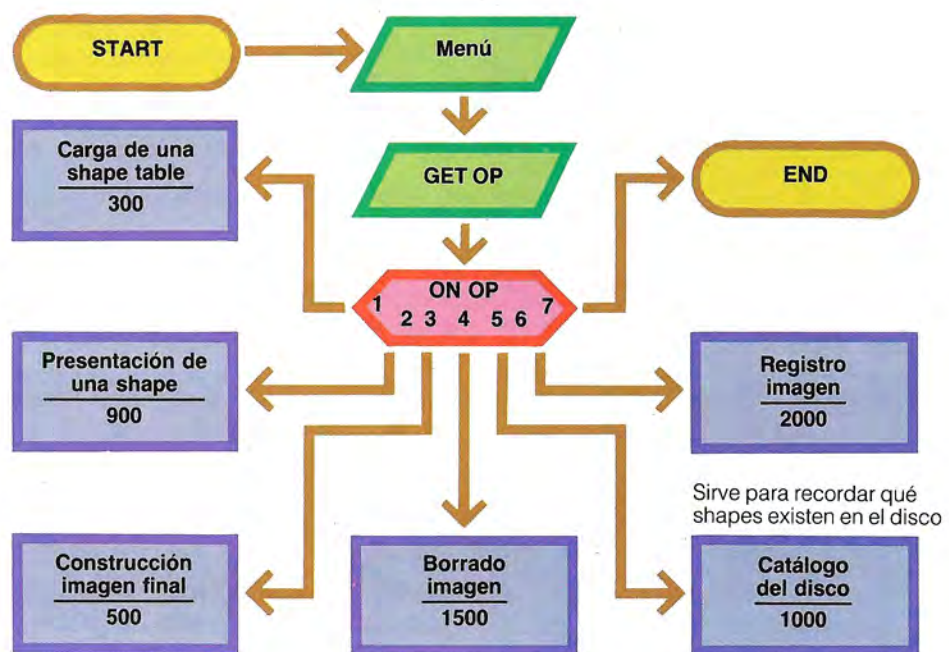
A continuación veremos las instrucciones utilizadas por los sistemas de la familia Apple.

DRAW. La sintaxis de la instrucción es

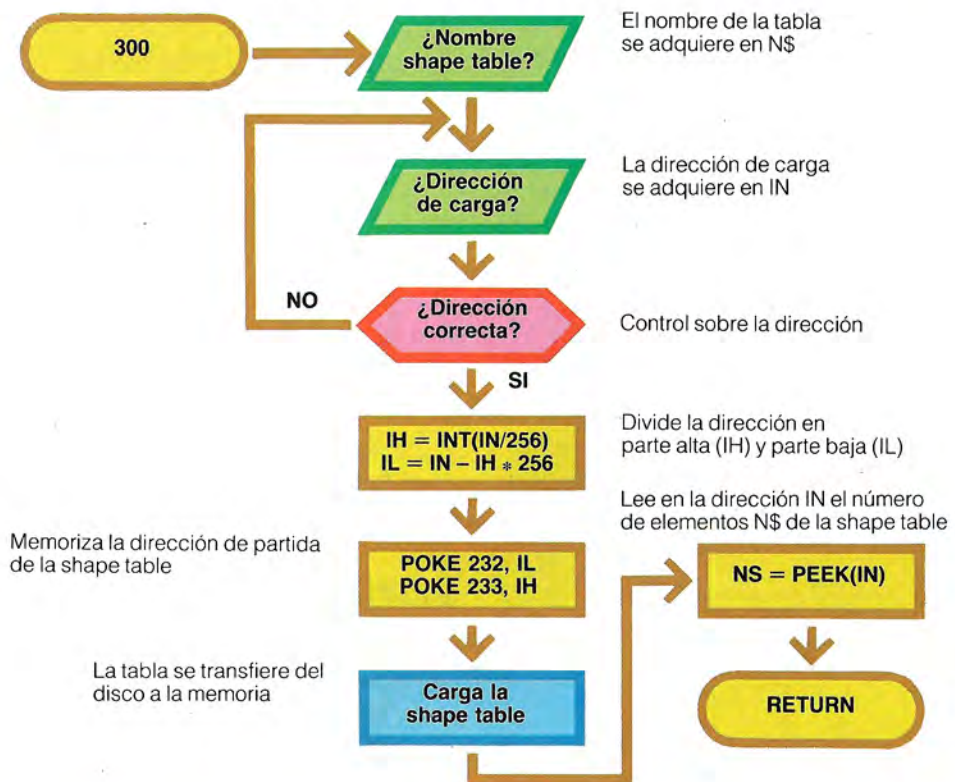
DRAW N AT X,Y

donde

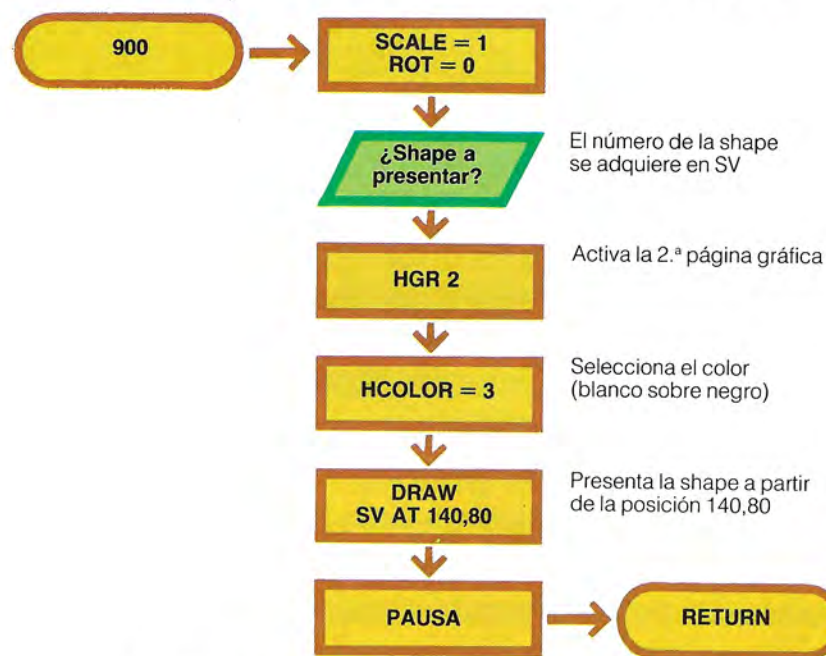
SHAPE LOADER



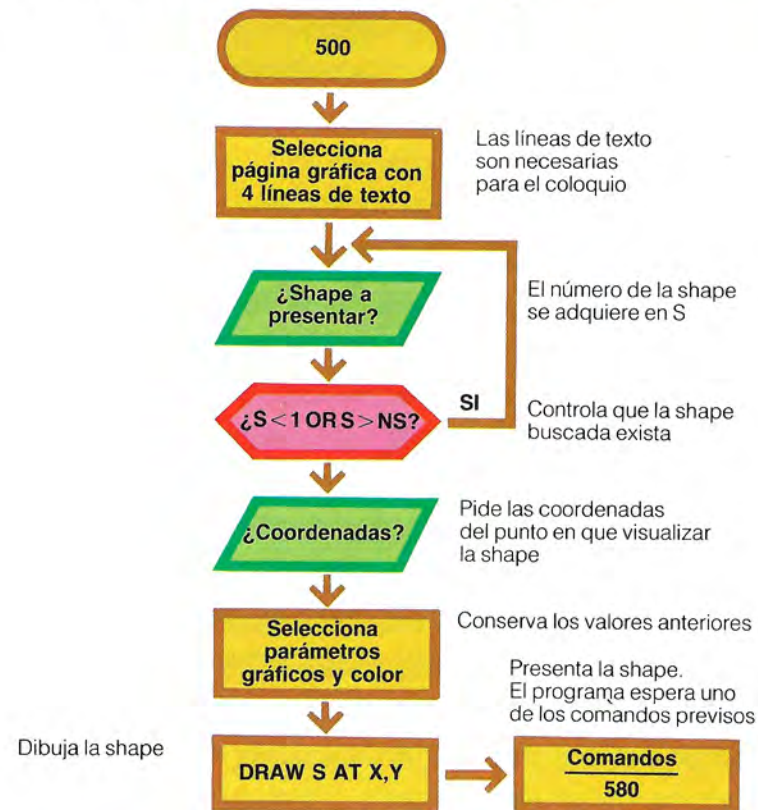
SUBROUTINA DE CARGA DE UNA SHAPE TABLE



SUBROUTINA DE PRESENTACION DE UNA SHAPE



Subrutina de construcción de la imagen final



COMANDOS PREVISTOS POR LA SUBROUTINA 580

Comando	Significado
/	Elimina ventana texto
?	Presenta ventana texto
I	Desplaza la shape hacia arriba un paso
J	Desplaza la shape a la izquierda un paso
K	Desplaza la shape a la derecha un paso
M	Desplaza la shape abajo un paso
R	Rota la shape 90° en sentido horario
S	Amplía la shape
CTRL + R	Rota la shape 90° en sentido antihorario
CTRL + S	Reduce la shape
C	Borra
ESC	Vuelve al programa principal

La subrutina 580 permite componer las diversas shape de la tabla cargada en memoria hasta formar un dibujo complejo.

SHAPE LOADER

```

10 REM *****
20 REM *
30 REM * SHAPE LOADER *
40 REM *
50 REM *****
60
:
70 REM MENU
80
:
90 TEXT
: HOME
100 H1 = PEEK (115) + PEEK (116) * 25
:
110 HTAB 18
: INVERSE
: PRINT "MENU"
120 NORMAL
: VTAB 6
130 PRINT "RECUPERACION DE UNA SHAPE
TABLE...1"
: PRINT
140 PRINT "DISPLAY DE UNA SHAPE.....
....2"
: PRINT
150 PRINT "CONSTRUCCION IMAGEN FINAL....
....3"
PRINT
160 PRINT "BORRADO IMAGEN.....
....4"
: PRINT
170 PRINT "CATALOGO DEL DISCO....
....5"
: PRINT
180 PRINT "GRABACION IMAGEN....
....6"
: PRINT
190 PRINT "FIN.....
....7"
200 VTAB 21
210 INPUT "QUE OPCION DESEA? ":
OP#

```

```

220 OP = VAL (OP#)
: IF OP < 1 OR OP > 7 THEN 90
230 ON OP GOSUB 300,900,500,1500,1000
,2000
240 IF OP < > 7 THEN 90
250 HOME
: END
300
:
310 REM CARGA SHAPE TABLE
320
:
330 TEXT
: HOME
340 INPUT "NOMBRE SHAPE TABLE? ":N#
345 IF ASC (LEFT# (N#,1)) < 65 OR
ASC (LEFT# (N#,1)) > 90
THEN 330
350 PRINT
: PRINT
360 PRINT "DIRECCION DE CARGA
"
: PRINT "(24576-"HI.")"
370 VTAB 4
: HTAB 26
: PRINT SPC (14)
380 VTAB 4
: HTAB 26
: INPUT " ".IN#
390 IN = VAL (IN#)
400 IF IN < 24576 OR IN > HI THEN 370
:
410 IH = INT (IN / 256)
IL = IN - IH * 256
420 POKE 232,IL
: POKE 233,IH
425 VTAB 16
: INVERSE
: PRINT "INSERTE UN DISCO Y PULSE
<RETURN>":
: NORMAL
: GET R#
427 PRINT CHR# (4)
430 PRINT CHR# (4) "BLOAD".N#;"A";IN
:
435 NS = PEEK (IN)
440 RETURN
500
:
505 REM CONSTRUCCION IMAGEN
510
:
515 IF NS = 0 THEN HOME
: VTAB 12
: FLASH
: PRINT "NO HAY TABLAS EN MEMO
: RIA";
: NORMAL
: GET A#
: RETURN
520 POKE - 16297,0
: POKE - 16300,0
: POKE - 16301,0
: POKE - 16304,0
: POKE 230,32
525 HOME
: VTAB 21
530 PRINT "SHAPE A PRESENTAR (1-"
;NS.")";
: INPUT " ? ";S#
535 IF S# = "" THEN RETURN
540 S = VAL (S#)
545 IF S < 1 OR S > NS THEN 525

```

```

546 VTAB 23
: INPUT "COORDENADAS (X,Y) ? ",X#,Y#
: X = VAL (X#)
: Y = VAL (Y#)
547 IF X < 0 OR X > 279 OR Y < 0
OR Y > 191 THEN 546
550 RO = 0
: SC = 1
555 X1 = X
: Y1 = Y
: R1 = RO
: S1 = SC
557 HCOLOR= 3
560 SCALE= SC
: ROT= RO
565 DRAW S AT X,Y
570 HOME
: VTAB 23
575 INVERSE
: PRINT "COMANDOS: I,J,K,MR,^R S,
^S /,? C ESC";
: NORMAL
580 HTAB 1
: VTAB 21
: PRINT SPC( 79)
585 HTAB 1
: VTAB 21
590 PRINT "SHAPE=";
: INVERSE
: PRINT S;
: NORMAL
595 PRINT " ESCALA=";
: INVERSE
: PRINT SC;
: NORMAL
600 PRINT " ROT=";
: INVERSE
: PRINT RO;
: NORMAL
605 PRINT " X=";
: INVERSE
: PRINT X;
: NORMAL
610 PRINT " Y=";
: INVERSE
: PRINT Y
: NORMAL
615 VTAB 1
: HTAB 1
: GET T#
620 IF T# = "/" THEN POKE - 16302,0
625 IF T# = "?" THEN POKE - 16301,0
630 IF T# = "I" THEN Y = Y - 1
: IF Y < 0 THEN Y = 191
635 IF T# = "J" THEN X = X - 1
: IF X < 0 THEN X = 279
640 IF T# = "K" THEN X = X + 1
: IF X > 279 THEN X = 0
645 IF T# = "M" THEN Y = Y + 1
: IF Y > 191 THEN Y = 0
650 IF T# = "R" THEN RO = RO + 1
: IF RO > 255 THEN RO = 255
655 IF T# = CHR# (18) THEN RO = RO
- 1
: IF RO < 0 THEN RO = 0
660 IF T# = "S" THEN SC = SC + 1
: IF SC > 255 THEN SC = 255
665 IF T# = "C" THEN GOSUB 1800
670 IF T# = CHR# (19) THEN SC = SC
- 1
: IF SC < 1 THEN SC = 1

```

```

675 IF T# = CHR# (27) THEN RETURN
677 HCOLOR= 0
680 SCALE= S1
: ROT= R1
685 DRAW S AT X1,Y1
687 HCOLOR= 3
690 SCALE= SC
: ROT= RO
: DRAW S AT X,Y
695 S1 = SC
: R1 = RO
: X1 = X
: Y1 = Y
700 GOTO 580
900
:
905 REM DISPLAY DE UNA SHAPE
910
:
915 IF NS = 0 THEN HOME
: VTAB 12
: FLASH
: PRINT "NO HAY TABLAS EN MEMO
RIA";
: NORMAL
: GET A#
: RETURN
920 TEXT
: HOME
925 SCALE= 1
: ROT= 0
930 VTAB 12
: HTAB 1
935 PRINT "SHAPE A PRESENTAR (1-"
;NS;") ";
: INPUT SV#
: SV = VAL (SV#)
940 IF SV < .1 OR SV > NS THEN 920
945 HGR2
: HCOLOR= 3
950 DRAW SV AT 140,80
955 GET R#
960 RETURN
1000
:
1010 REM CATALOG
1020
:
1030 HOME
: PRINT CHR# (4); "CATALOGO"
1040 GET A#
: RETURN
1075 PRINT CHR# (4)
1500
:
1510 REM BORRADO
1520
:
1530 HOME
: VTAB 12
: INPUT "SEGURO (S/N) ? ",R#
1540 IF R# < > "S" AND R# < > "N"
THEN 1530
1550 IF R# = "S" THEN HGR
1560 RETURN
1800
:
1810 REM MODO BORRA
1820
:
1830 XO = X
: YO = Y
: X = 140

```

```

: Y = 80
1835 VTAB 23
: HTAB 36
: PRINT "C"
1840 HCOLOR= 3
: HPLOT X,Y
1850 VTAB 1
: HTAB 1
GET A$
1855 HCOLOR= 0
: HPLOT X,Y
1860 IF A$ = "I" THEN Y = Y - 1
: IF Y < 0 THEN Y = 191
1870 IF A$ = "M" THEN Y = Y + 1
: IF Y > 191 THEN Y = 0
1880 IF A$ = "J" THEN X = X - 1
: IF X < 0 THEN X = 279
1890 IF A$ = "K" THEN X = X + 1
: IF X > 279 THEN X = 0
1900 IF A$ = "C" THEN 1920
1910 GOTO 1840
1920 HCOLOR= 3
: X = X0
: Y = Y0
1930 VTAB 23
: INVERSE
1940 PRINT "COMANDOS I,J,K,M R,^R S,
AS /,? C ESC";
: NORMAL
1950 RETURN
2000
:
2010 REM SALVA IMAGEN
2020
:
2030 HOME
: VTAB 12
2040 INPUT "CON QUE NOMBRE? ";NM$
2050 IF ASC ( LEFT$( NM$,1)) < 65
OR ASC ( LEFT$( NM$,1)) > 90
THEN 2030
2060 VTAB 14
: PRINT "DE ACUERDO! ";NM$".PIC"
2070 VTAB 16
: INVERSE
: PRINT "INSERTE UN DISCO Y PULSE
<RETURN>";
: NORMAL
: GET R$
2080 PRINT CHR$( 4)
2090 PRINT CHR$( 4), "BSAVE":NM$;".PIC,
A$2000,L$1FFF"
2100 PRINT CHR$( 4)
2110 RETURN

```

VARIABLES UTILIZADAS EN EL PROGRAMA SHAPE LOADER

HI	= Límite de la memoria reservada a los gráficos
IN	= Dirección de partida de la shape table
N\$,NM\$	= Nombre de la shape table
NS	= Número de la shape en la tabla
R1,RO	= Valores de ROT
S\$	= Shape en curso
S1,SC	= Valores de SCALE
X,Y,X0,Y0,X1,Y1	= Coordenadas de pantalla

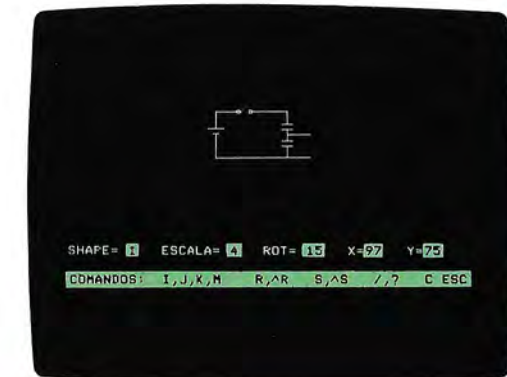
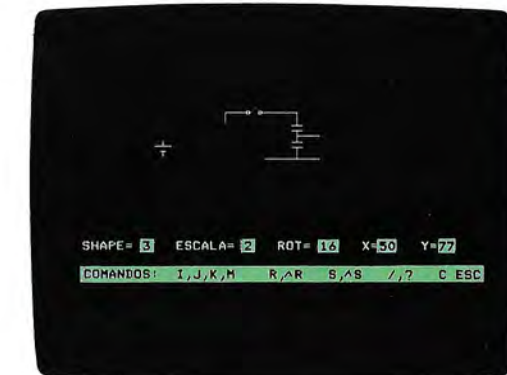
EJEMPLO DE APLICACION DEL PROGRAMA SHAPE LOADER

La secuencia fotográfica ilustra una aplicación del programa Shape Loader para la construcción de una imagen compuesta que utiliza algunas figuras elementales (shapes) creadas y memorizadas mediante el Shape Editor.

El programa ha presentado el menú del procedimiento. El usuario introducirá la petición de carga desde el disco de una shape table (tabla de las figuras) que contiene algunos símbolos eléctricos.

En esta fase se ha entrado en la rutina de composición de la imagen final. Abajo se ve el coloquio con el menú de los comandos; en el centro de la pantalla, el dibujo en composición. El elemento de la izquierda es la shape que está por posicionarse.

La construcción de la figura compuesta ahora está completa. El usuario la memorizará en el disco dándole un nombre específico que permitirá recargarla cada vez que sea necesario.



N es el número de la figura (perteneciente a la tabla actualmente en memoria) que se quiere presentar

X,Y son las coordenadas a partir de las cuales debe empezar el trazado de la figura.

Por ejemplo, la instrucción

DRAW 2 AT 30,50

presenta la figura número 2 empezando su dibujo desde el punto de coordenadas X = 30 e Y = 50. Si las coordenadas no se especifican, la

figura empieza por el último punto llamado y, por tanto, la instrucción también es válida en la forma DRAW N, que presenta la figura N a partir de las últimas coordenadas utilizadas en una instrucción anterior DRAW o HPLOT.

XDRAW. Tiene una forma idéntica a la anterior:

XDRAW AT X,Y

especifica el borrado de la figura número N posicionada en las coordenadas X,Y.

ROT. La sintaxis de la instrucción es sencilla:

ROT M

con M un valor numérico que expresa el ángulo de rotación, generalmente en grados sexagesimales. El intervalo de validez del parámetro M depende del valor definido en la anterior instrucción SCALE. Si el factor de escala se implanta igual a 1, el parámetro M sólo puede asumir uno entre los valores: 0 = 0°, 16 = 90°, 32 = 180°, 48 = 270°.

Y viceversa, si el factor de escala es superior a 4, son posibles 64 rotaciones diferentes equidistantes con respecto a los valores anteriores (estos valores sólo son un ejemplo ligado al hardware particular y no constituyen una regla).

SCALE. Debe utilizarse antes de cualquier otra instrucción de este grupo porque define la escala que deberá tener el dibujo.

La sintaxis es

SCALE = Q

donde el parámetro numérico Q indica cuántas veces debe trazarse cada vector gráfico sencillo. Por ejemplo, SCALE = 1 presenta una figura trazando cada vector gráfico una sola vez, mientras que SCALE = 2 duplica las dimensiones trazando dos veces cada vector sencillo.

En algunas máquinas, la preparación de un dibujo por medio de desplazamientos elementales puede obtenerse con una cadena de comando. Generalmente, la palabra clave todavía es DRAW, pero con sintaxis y significados diferentes, aunque análogos.

Por tanto, existen dos formas de esta instrucción. En la primera, los comandos de desplazamiento están contenidos en una cadena asociada a la instrucción; en la segunda, los comandos están memorizados en una tabla residente en memoria, y la instrucción acepta como parámetro el número correspondiente a la figura a representar:

DRAW: "Cadena de comandos"

DRAW: N con N = número de la figura a presentar

En la primera forma (cadena) es necesario explicitar algunos símbolos que representan las acciones elementales a realizar. A continuación se ilustran los correspondientes al sistema Olivetti M20.

Los parámetros previstos son:

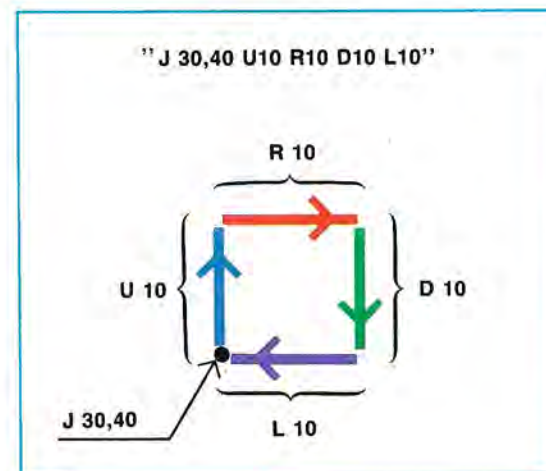
M DX,DY Desde el punto actual de coordenadas X,Y se posiciona en el punto de coordenadas X + DX e Y + DY
 J X,Y se desplaza al punto X,Y
 U DY se desplaza hacia arriba en la cantidad DY
 L DX se desplaza a la izquierda en DX
 R DX se desplaza a la derecha en DX
 C especifica el color con que debe trazarse la figura; si este parámetro no está presente en la instrucción, el sistema utiliza el último color empleado en una instrucción DRAW

En la figura de abajo se ha representado como ejemplo la cadena de comandos necesaria para trazar un cuadrado de lado 10 posicionado en 30,40 (vértice de abajo a la izquierda). La cadena contiene el direccionado en X = 30, Y = 40 (código J) y los cuatro desplazamientos necesarios para obtener los lados.

Tal como está presentada, la ilustración en realidad traza también una línea de la posición en que se encontraba el cursor al primer vértice del cuadrado, puesto que el comando J 30,40 se realiza con el trazo en visión. Para evitar que se presente este segmento no deseado es necesario hacer preceder el comando por la letra B, que impide la visualización. Por lo tanto, la forma exacta de la cadena es

"BJ 30, 40, U10 R10 D10 L10"

Espacios blancos entre un comando y el otro



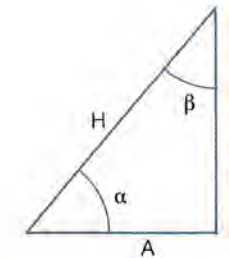
no son significativos. Para cada comando además es posible utilizar una de las siguientes opciones:

AND El color resultante se obtiene aplicando el operador AND entre el preexistente y el especificado en la cadena
 XOR, OR Tienen el mismo significado, pero con los operadores XOR y OR
 NOT Indica que el color de la figura es el complemento del color utilizado como fondo.

Gráficos tridimensionales

La preparación en tres dimensiones en la pantalla de vídeo requiere el uso de fórmulas matemáticas, a veces complejas, pero a continuación se presentarán las nociones básicas, útiles para la mejor comprensión del tema y para su aplicación a los problemas prácticos. Las formas más utilizadas son las de la trigonometría aplicada a la solución de los triángulos rectángulos. En la figura de abajo se ha representado una tabla que muestra cómo calcular los elementos desconocidos de un triángulo rectángulo.

RESOLUCION DE TRIANGULOS RECTANGULOS



$$B = H \cdot \sin(\alpha)$$

$$A = H \cdot \cos(\alpha)$$

$$\frac{B}{A} = \frac{\sin(\alpha)}{\cos(\alpha)} = \operatorname{tg}(\alpha)$$

Datos*	Cálculo de los elementos desconocidos
H, α	Se aplican las fórmulas anteriores
A, B	$\alpha = \operatorname{arctg}(B/A)$ $H = A/\cos(\alpha) = B/\sin(\alpha)$
A, α	$H = A/\cos(\alpha)$ $B = H \cdot \sin(\alpha) = A \cdot \frac{\sin(\alpha)}{\cos(\alpha)} = A \cdot \operatorname{tg}(\alpha)$
B, α	$H = B/\sin(\alpha)$ $A = H \cdot \cos(\alpha) = B \cdot \frac{\cos(\alpha)}{\sin(\alpha)} = \frac{B}{\operatorname{tg}(\alpha)}$
H, A	$\cos(\alpha) = A/H$ $\sin(\alpha) = \sqrt{1 - \cos^2(\alpha)}$ $B = H \cdot \sin(\alpha)$
H, B	$\sin(\alpha) = B/H$ $\cos(\alpha) = \sqrt{1 - \sin^2(\alpha)}$ $A = H \cdot \cos(\alpha)$

Todas las fórmulas pueden traducirse en lenguaje simbólico para ser insertadas en un programa.

Debe tenerse en cuenta que

– el ángulo α debe ser en radianes

– los nombres de las funciones son generalmente

$$\sin(\alpha) = \operatorname{SEN}(\operatorname{ALFA})$$

$$\cos(\alpha) = \operatorname{COS}(\operatorname{ALFA})$$

$$\operatorname{tg}(\alpha) = \operatorname{SEN}(\operatorname{ALFA})/\operatorname{COS}(\operatorname{ALFA})$$

$$\sqrt{\dots} = \operatorname{SQR}(\dots)$$

$$\operatorname{arctg}(B/A) = \operatorname{ATN}(B/A)$$

* El caso en que se conozca el ángulo β en lugar de α no se considera, porque $\alpha = 90 - \beta$; este caso no puede resolverse utilizando una de las fórmulas indicadas.

lo que pueden presentarse. Las funciones trigonométricas utilizadas, son

- seno = $\text{sen}(\alpha)$
- coseno = $\text{cos}(\alpha)$
- tangente = $\text{tg}(\alpha)$
- arcotangente = $\text{arctg}(\alpha)$

Dados la hipotenusa H y el ángulo α , para hallar los catetos basta con aplicar las fórmulas

- $A = H * \text{cos}(\alpha)$
- $B = H * \text{sen}(\alpha)$

En los lenguajes de programación como el Fortran y el Basic también hay las principales funciones trigonométricas; las instrucciones necesarias para realizar el cálculo anterior son

- $A = H * \text{COS}(\text{ALFA})$
- $B = H * \text{SEN}(\text{ALFA})$

con la única advertencia de transformar el ángulo a radianes, utilizando la expresión

$$\begin{aligned} \text{ángulo en radianes} &= \frac{\pi * \text{ángulo en grados}}{180} = \\ &= \frac{3.14 * \text{ángulo en grados}}{180} \end{aligned}$$

Obsérvese que en algunos lenguajes no hay la función $\text{tg}(\alpha)$, y que en este caso debe calcularse con la expresión

$$\text{TG}(\text{ALFA}) = \text{SEN}(\text{ALFA}) / \text{COS}(\text{ALFA})$$

En el programa, TG debe definirse como función de usuario (con la instrucción DEF...), después de haberla asimilado a una función intrínseca.

En la solución de los triángulos rectángulos puede encontrarse una complicación cuando entre los elementos conocidos no hay ángulos, sino lados. En estos casos, además del teorema de Pitágoras, pueden utilizarse las funciones trigonométricas inversas. Por ejemplo, conociendo los dos catetos, utilizando la función arcotangente (indicada con la sigla ATN), puede calcularse el ángulo así:

$$\text{ALFA} = \text{ATN}(B/A) \text{ con } A \text{ y } B \text{ catetos}$$

Normalmente, el arcotangente también está incluido entre las funciones intrínsecas del lenguaje de programación.

Las funciones de dos variables

Las funciones consideradas hasta ahora tienen la forma $Y = f(X)$. La variable dependiente Y es función de una sola variable independiente X. Las funciones de una variable pueden representarse sobre un plano cartesiano, puesto que cada punto está determinado por un par de valores (coordenadas respecto al sistema de referencia utilizado). Los procesos de estas funciones pueden visualizarse fácilmente haciendo coincidir los ejes de referencia con dos ejes de la pantalla vídeo. Sin embargo, también existen funciones de más variables. En particular se utilizan con frecuencia las de dos variables independientes, que asumen la forma $Y = f(X,Z)$, o también $Z = f(X,Y)$.

La simbología es la misma: el símbolo f indica una función general, o sea una ley que liga los valores de la variable dependiente a los de la variable independiente. Utilizar una u otra de las formas indicadas sólo depende del simbolismo adoptado; la primera [$Y = f(X,Z)$] tiene un aspecto más homogéneo con las formas vistas.

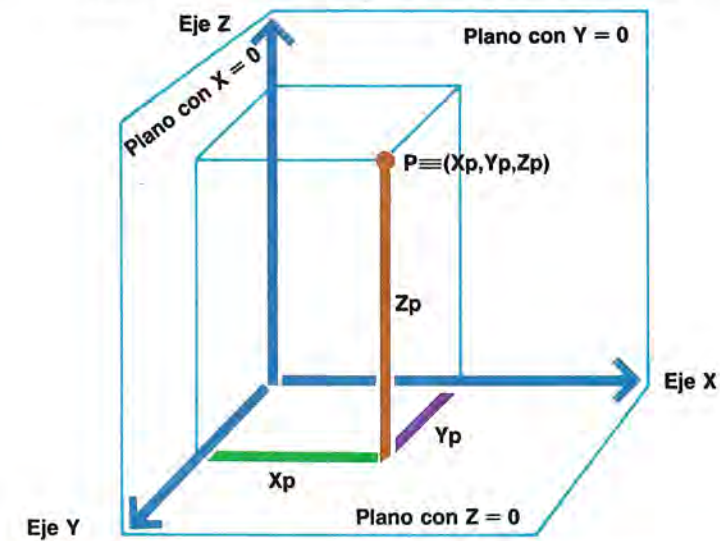
Una función de este tipo no puede representarse en un plano, puesto que debe indicar los valores de una tercera coordenada (Z). El sistema de referencia a adoptar debe prever tres ejes en lugar de los dos utilizados en el plano. En la figura de arriba de la página siguiente se ha representado un sistema de referencia con los tres ejes X, Y, Z. Cada punto del espacio está identificado por las coordenadas correspondientes a cada uno de los tres ejes.

La dificultad más obvia e inmediata que afecta a la representación de una función de dos variables es la ausencia del tercer eje en la pantalla de vídeo. Los métodos de representación requieren por tanto la simulación de la profundidad; es decir, deben desarrollarse algoritmos análogos a los utilizados por un dibujante para construir una perspectiva.

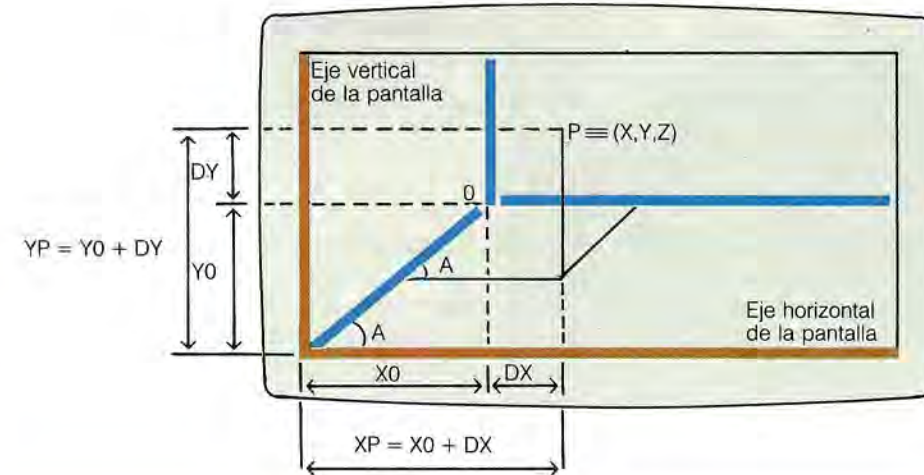
Los métodos que se expondrán a continuación no constituyen verdaderamente aplicaciones de gráficos tridimensionales; en cambio se trata de simulaciones de tridimensionalidad adecuadas a muchos objetivos prácticos, pero limitadas por el ambiente hardware en que residen.

Para realizar un verdadero software gráfico tridimensional son necesarias máquinas de elevadas potencialidades y con componentes hardware dedicados a este objeto, cuyo coste puede ser incluso 100 veces superior al de un ordenador personal.

SISTEMA DE REFERENCIA EN EL ESPACIO



REPRESENTACION DE UN SISTEMA DE REFERENCIA ESPACIAL EN LA PANTALLA VIDEO



Representación tridimensional. El procedimiento utilizado para representar en el monitor un punto P genérico del espacio se esquematiza en la figura de arriba.

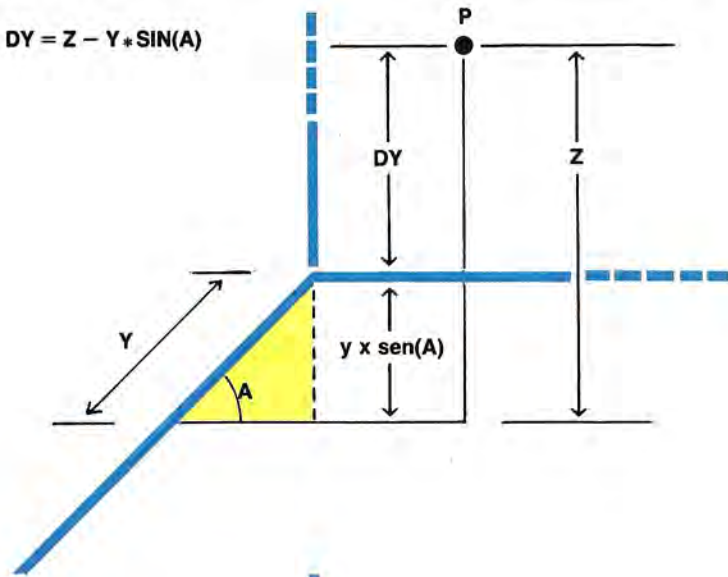
En un punto interior de la pantalla se sitúa el origen del sistema de referencia, indicado con 0. Desde el origen parten los tres ejes cartesianos. Por sencillez supondremos que el origen de los ejes de la pantalla está abajo a la izquierda (en algunos sistemas está arriba a la izquierda).

El punto P genérico determinado en el espacio por las coordenadas X, Y, Z tiene, con respecto a la pantalla (sólo desde el punto de vista gráfico), las coordenadas XP e YP. Estas coordenadas pueden obtenerse sumando respectivamente las cantidades DX y DY a las coordenadas del origen de los ejes. Por tanto, se obtiene

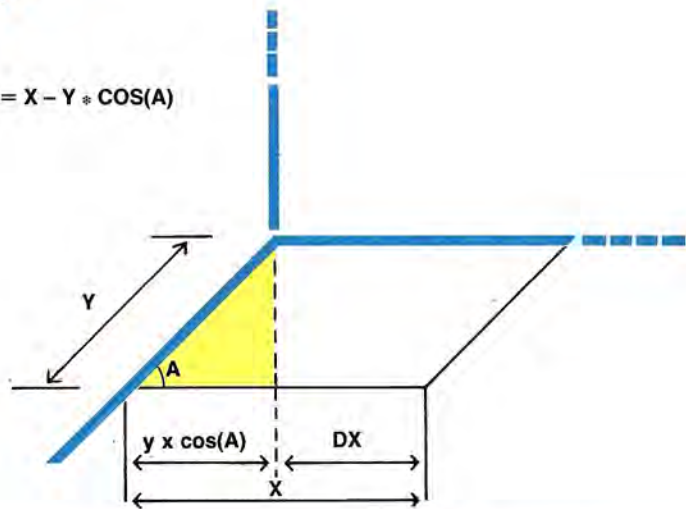
$$\begin{aligned} X_P &= X_0 + DX \\ Y_P &= Y_0 + DY \end{aligned}$$

CALCULO DE LAS COORDENADAS DE PANTALLA

$$DY = Z - Y * \text{SEN}(A)$$



$$Dx = X - Y * \text{COS}(A)$$



Conocidos los valores DX y DY pueden calcularse las coordenadas de pantalla para todos los puntos de la función tridimensional. En último análisis, el problema se reduce al cálculo de las cantidades DX y DY. El primer paso es el de calcular el ángulo A, utilizando las fórmulas trigonométricas anteriores (para brevedad, el ángulo se ha indicado con la sola letra A). El valor de A se determina con las coordenadas X0, Y0, observando que éstas construyen los catetos de un triángulo rectángulo que tiene como hipotenusa el segmento que une el origen de la referencia de la pantalla con el punto 0.

En consecuencia, se obtiene $A = \text{ATN}(Y0/X0)$. Determinado este ángulo puede procederse al cálculo de DX y DY, utilizando siempre las reglas de solución de los triángulos rectángulos. Así se obtiene

$$DY = Z - Y * \text{SEN}(A)$$

$$DX = X - Y * \text{COS}(A)$$

Para comprender las fórmulas anteriores deben observarse las figuras que aparecen en esta página, sin considerar la tridimensionalidad del sistema de referencia.

Determinados de la manera vista los parámetros y las fórmulas necesarias, para dibujar una función tridimensional cualquiera es necesario:

- 1 / Asignar valores arbitrarios a las dos variables independientes X y Z
- 2 / Calcular el correspondiente valor de Y utilizando la expresión matemática de la función
- 3 / Calcular las coordenadas XP e YP en puntos de pantalla por medio de las fórmulas

$$YP = X0 + DX$$

$$XP = Y0 + DY$$

o con las análogas

$$XP = X0 + (X - Y * \text{COS}(A))$$

$$YP = Y0 + (Z - Y * \text{SEN}(A))$$

habiendo sustituido

$$DX = X - Y * \text{COS}(A)$$

$$DY = Z - Y * \text{SEN}(A)$$

Obsérvese que las coordenadas XP, YP en puntos de pantalla no tienen nada que ver con las coordenadas X, Y de los puntos de las curvas. Por tanto, si la función está en la forma $Z = f(X, Y)$ o en la forma $Y = f(X, Z)$, todas las fórmulas continúan siendo válidas, intercambiando las coordenadas utilizadas en los cálculos: las variables independientes en este caso son X, Y en lugar de X, Z, y la función permite el cálculo de Z en lugar del de Y. La forma $Z = f(X, Y)$ es la más usual, puesto que es la que mejor se presta para la interpretación del gráfico. El diagrama de flujo de un programa para la representación de curvas en tres dimensiones con el método expuesto se representa en las figuras de las páginas 1638 y 1639. Para simplificar se han omitido todos los controles y los eventuales cambios de escala. El bucle más externo se ha realizado sobre la coordenada Y (entendida como coordenada de la función), y el más interno se ha realizado sobre la X. Esto equivale a presentar el proceso de la función por secciones sucesivas paralelas al plano XZ. Por ejemplo, supongamos que el valor inicial de la coordenada Y sea cero, o sea $Y1 = 0$. El bucle más interno (en la X) mostraría todos los puntos de la curva que tienen coordenadas $Y = 0$, o sea el proceso de la porción de la curva que está sobre el plano $Y = 0$. Terminado el bucle interno se tiene un incre-

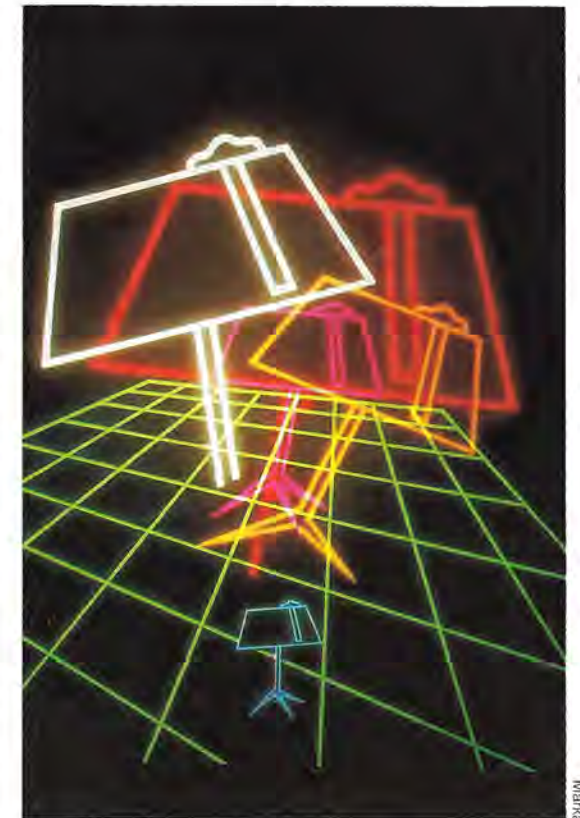
mento en el más externo, por lo que la Y pasa a un valor siguiente que podría ser $Y = 1$. En este caso, el bucle interno presenta el proceso de la función en el plano $Y = 1$ (paralelo al anterior). Siguiendo de esta manera, siempre incrementando el bucle externo en 1, se tiene el gráfico tridimensional de la función como superposición de los gráficos planos, cada uno de los cuales muestra el proceso de un plano particular ($Y = 0, Y = 1, Y = 2, \dots$).

Estas representaciones en planos paralelos son secciones de la figura sólida representadas por la curva; el mecanismo se ha esquematizado en la figura de arriba de la página 1640. El listado del programa se ha representado en las páginas 1640 a 1642.

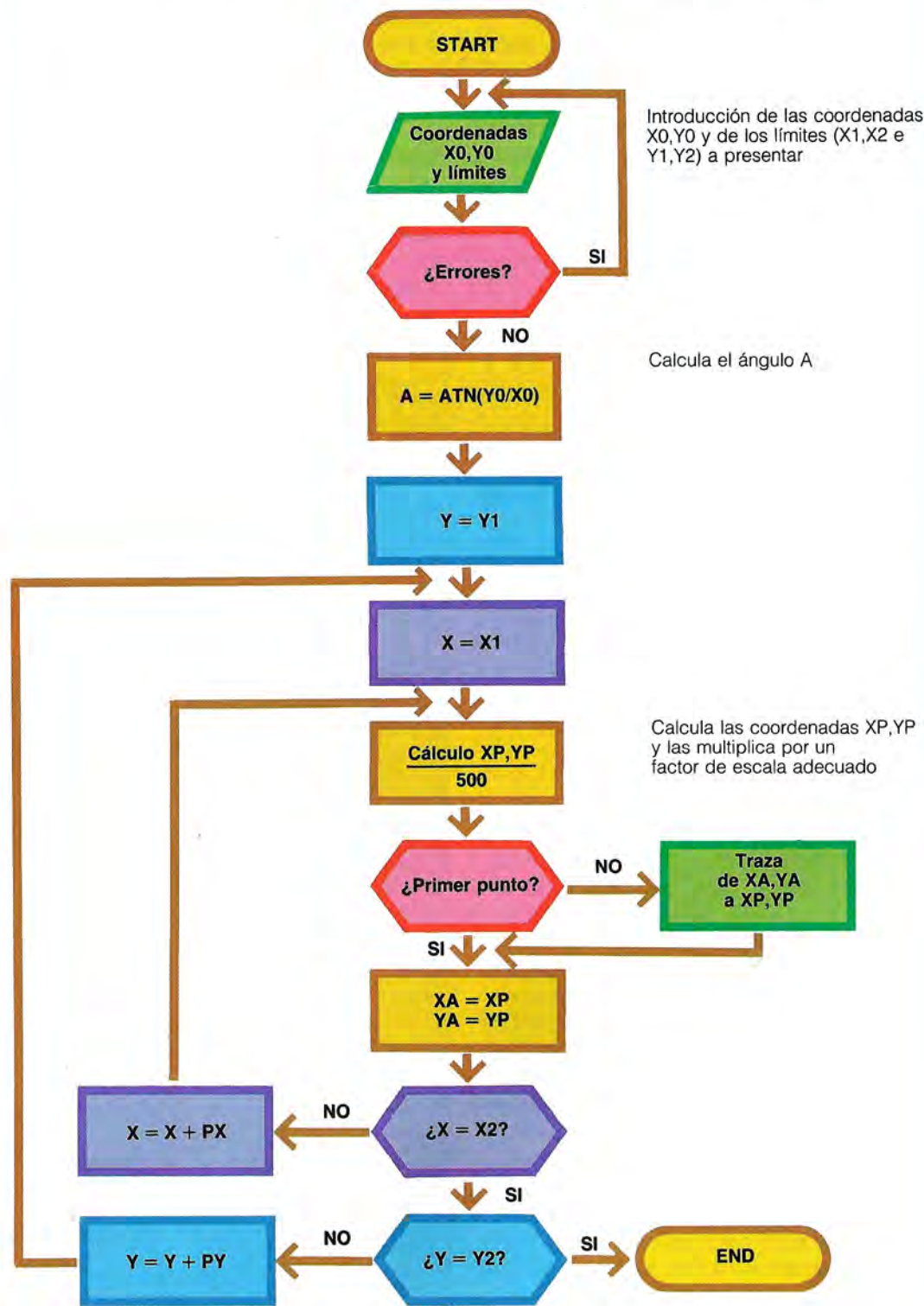
Los problemas gráficos en tres dimensiones.

Debe distinguirse entre la necesidad real de gráficos en tres dimensiones y la simple representación tridimensional; los dos puntos de vista son notablemente diferentes y necesitan un hardware y un software de potencias y costes

Imagen obtenida con un programa de gráficos tridimensionales.



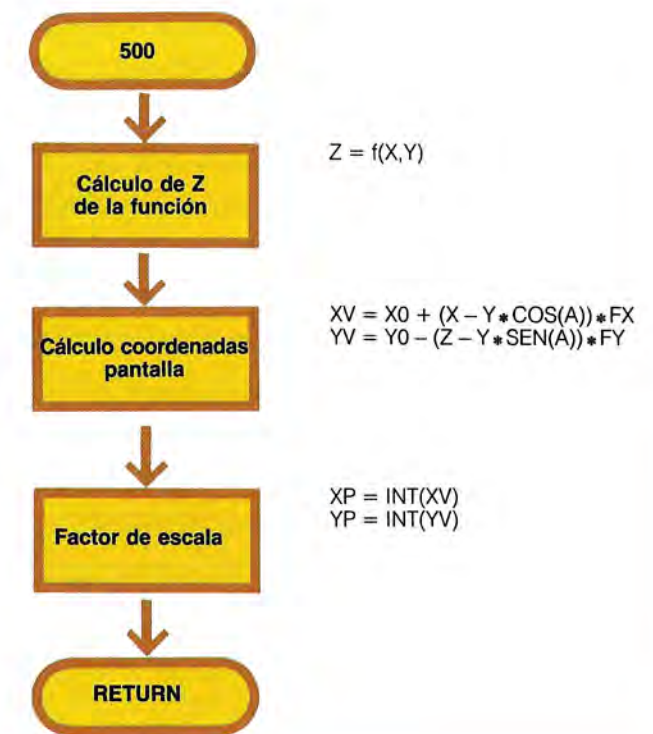
PRESENTACION DE UNA FUNCION DE DOS VARIABLES



totalmente diferentes. Los gráficos en tres dimensiones requieren la memorización de los atributos de cada punto del espacio que constituye el sólido a representar; de esta manera pueden obtenerse proyecciones, secciones y cualquier otro proceso gráfico del objeto. Sin embargo, un software de este tipo sólo puede residir en máquinas de potencias elevadas tanto en capacidad de memoria como en la resolución del vídeo o de los demás periféricos (por ejemplo, el plotter). Por tanto, se trata de funciones para las cuales hacen falta programas muy sofisticados e importantes recursos financieros. Sin embargo, en la representación tridimensional, el problema queda muy simplificado y puede reducirse a un sencillo proceso de una vista del objeto que proporcione la sensación de tridimensionalidad, sin que hagan falta máquinas y programas excesivamente sofisticados. En la figura de la página 1642 se muestra el método utilizado en un programa de demostración en que la tercera dimensión se obtiene trasladando la figura de base (vista frontal del objeto)

con una inclinación de 45° (de esta manera, los desplazamientos de cada punto son iguales para los dos ejes). Determinando los vértices de la figura base con los números de 1 a 6, la sensación del espesor se obtiene trazando la misma figura con las coordenadas de los vértices incrementadas en la cantidad D: obtenido el contorno trasladado, después es necesario unir los puntos homólogos. Sin embargo, continúa existiendo el problema de borrar (o mejor de no trazar) los segmentos de la vista trasladada que quedan cubiertos por la de base. Sobre este tema existen numerosos algoritmos, que también pueden utilizarse en la presentación de las funciones. Suelen ser soluciones muy especializadas y difícilmente generalizables para los dibujos en los que las líneas no son funciones matemáticas precisadas. O sea, el software debe poder detectar qué líneas quedan ocultas y evitar su presentación. El método utilizado en el programa que presentaremos se basa en un análisis realizado sólo con respecto a la figura de base, y en algunos casos puede dar resultados no correctos.

SUBROUTINA DE CALCULO DE LAS COORDENADAS DE PANTALLA



ESQUEMA DEL METODO DE REPRESENTACION

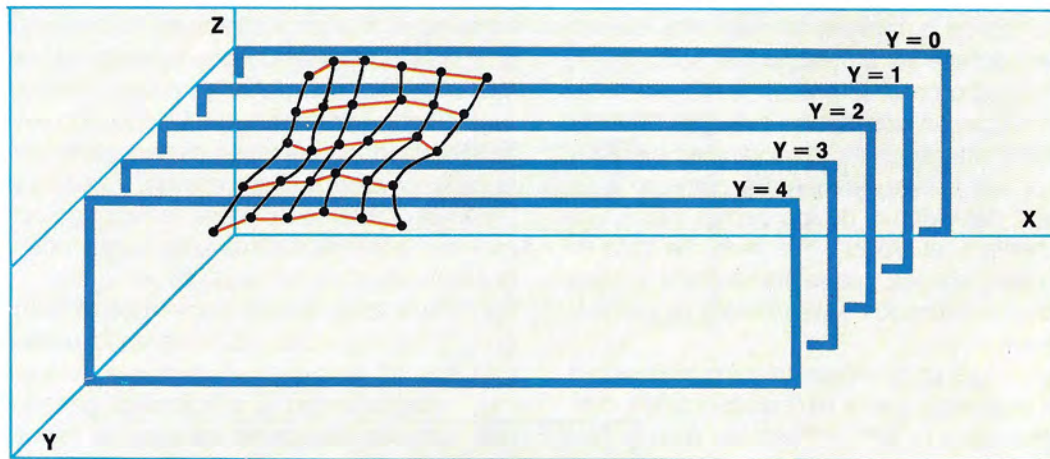


GRAFICO DE UNA FUNCION DE DOS VARIABLES

```

10  REM *****
20  REM *
30  REM *
40  REM * GRAFICOS DE 3 DIMENSIONES *
50  REM *
60  REM *
70  REM *****
80
:
90
:
100 REM -----
110 REM LIMITES DE LA PANTALLA
120 REM -----
130 XN = 0
: XM = 279
: YN = 0
: YM = 191
140 REM -----
150 REM FACTORES DE ESCALA
160 REM -----
170 FX = 5
: FY = 10
180 REM -----
190 REM DENSIDAD DE LOS PUNTOS
200 REM -----
210 PX = .2
: PY = .2
220 REM -----
230 REM COORDENADAS CENTRO
240 REM -----
250 X0 = 129
260 Y0 = 61
270 REM -----
280 REM INPUT DE LOS DATOS
290 REM -----
300 TEXT
: HOME
310 VTAB 12
320 INPUT "INTERVALO EJE X? ";X1,X2
330 VTAB 14
340 INPUT "INTERVALO EJE Y? ";Y1,Y2

```

```

350 A = ATN (Y0 / X0)
: REM ANGULO
360 HGR2
: HCOLOR= 3
370 GOSUB 2000
380
:
390
:
400 REM *****
410 REM **
420 REM ** BUCLE PRINCIPAL **
430 REM **
440 REM *****
450
:
460
:
470 FLAG = 1
480 FOR Y = Y1 TO Y2 STEP PY
490 FOR X = X1 TO X2 STEP PX
500 REM -----
510 REM CALCULO XP,YP
520 REM -----
530
:
540
:
550 Z = ( SEN (X) / X) * (5 * SEN (Y)
)
560
:
570
:
580 DX = X - Y * COS (A)
590 DY = Z - Y * SEN (A)
600 XV = Y0 + DX * FX
610 YV = Y0 - DX * FY
620 XP = INT (XV)
630 YP = INT (YV)
640 REM -----
650 REM DIBUJA GRAFICO
660 REM -----
670 IF FLAG THEN FLAG = 0
: GOTO 690
680 HPLLOT XA,YA TO XP,YP
690 XA = XP
YA = YP
700 NEXT X
710 FLAG = 1
720 NEXT Y
730
:
740
:
750 REM *****
760 REM ** FIN BUCLE **
770 REM *****
780
:
790
:
800 GET I$
810 TEXT
: HOME
820 END
2000 REM -----
2010 REM EJES CARTESIANOS
2020 REM -----
2030 HPLLOT XN,YN TO XM,YN TO XM,YM
TO XN,YM TO XN,YN

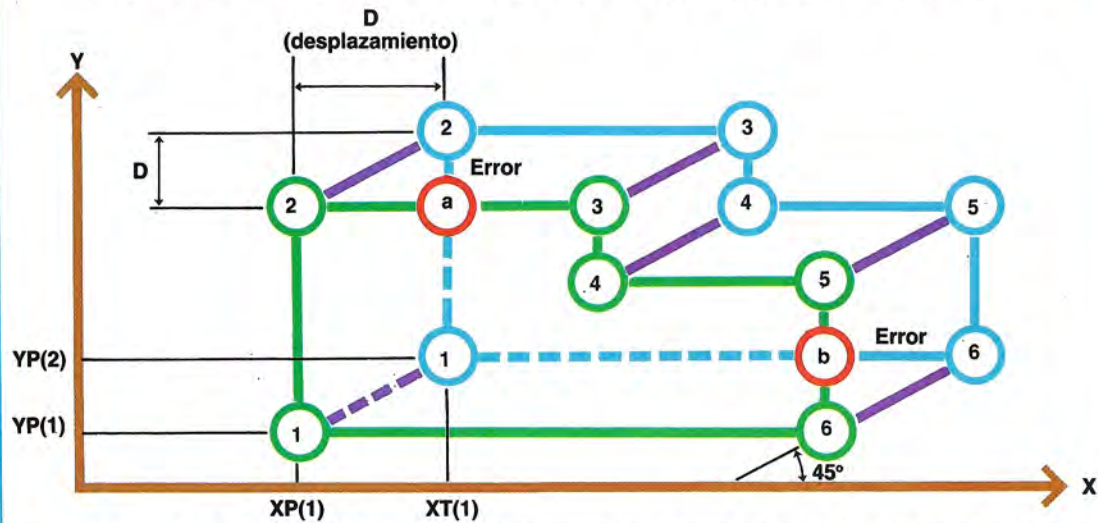
```

```

2040 HPL0T X0,Y0 TO X0,YN
2050 HPL0T XN,Y0 TO XM,Y0
2060 HPL0T X0,Y0 TO XN,YM
2065 HCOLOR = 0
2070 FOR I = X0 - 3 TO 0 STEP - 3
2080 HPL0T I,Y0
2090 NEXT
2100 HCOLOR = 3
2110 RETURN

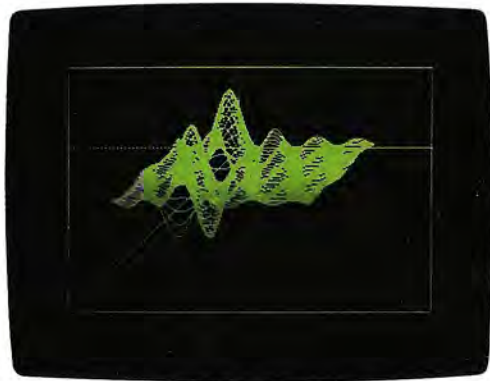
```

METODO UTILIZADO PARA LA REPRESENTACION TRIDIMENSIONAL



Los segmentos que llevan el rótulo Error se trazan porque el programa no está completo con todas las funciones necesarias.

- Contorno de referencia (es el primero en ser dibujado)
- Contorno trasladado
- Líneas de unión entre dos contornos (espesor)
- Vértices de base
- Vértices de base trasladados
- Puntos de intersección



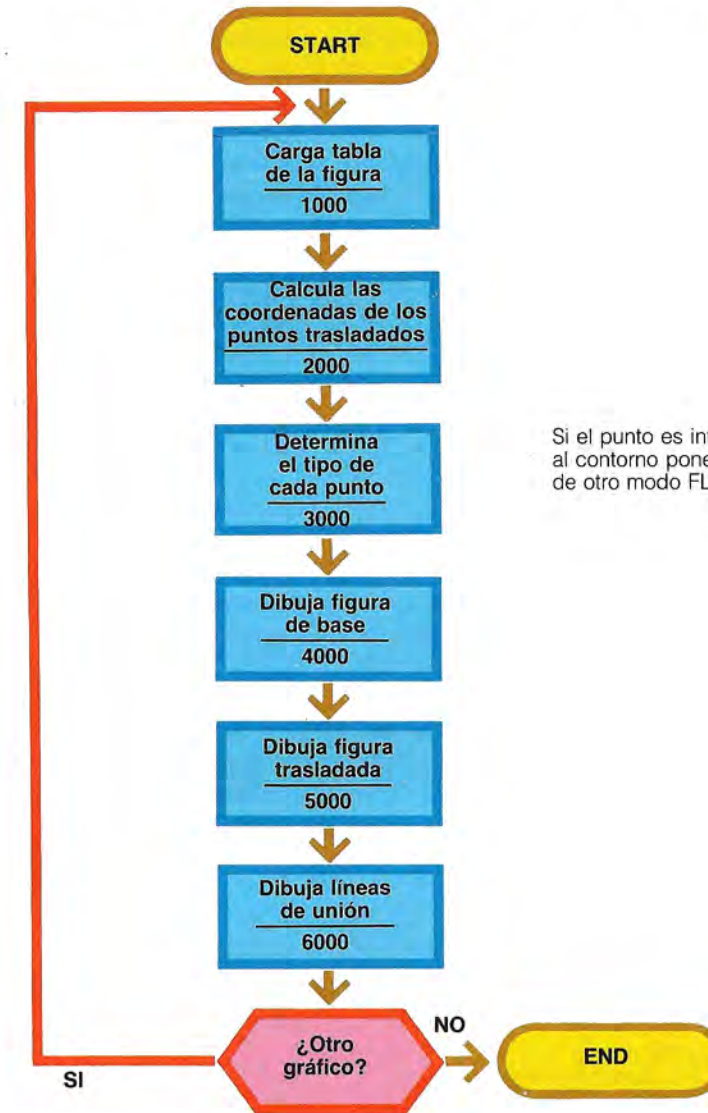
Resultado final de la presentación de una función de dos variables con el programa descrito.

Por tanto, debe considerarse únicamente como un ejemplo de aproximación al problema y no un método generalizado.

En la página siguiente se ha representado el diagrama de flujo del programa. Las funciones realizadas son:

- Adquisición de las coordenadas de los vértices de la figura de base (subrutina 1000) que por traslación origina la vista tridimensional
- Cálculo de las coordenadas de los puntos trasladados (subrutina 2000). El método consiste simplemente en incrementar todas las coordenadas con la cantidad S. Por brevedad no se han utilizado los métodos más rigurosos (de perspectiva).

DIAGRAMA DEL PROGRAMA DE SIMULACION TRIDIMENSIONAL



Si el punto es interior al contorno pone FL(·) = 1, de otro modo FL(·) = 0

Los diagramas de flujo de detalle que siguen se han simplificado en algunos puntos con respecto al listado

Principales variables utilizadas

XP(21)	}	Coordenadas de cada punto de la figura de base
XP(1)		
XT(21)		
YT(21)	}	Coordenadas de la figura trasladada
FL(20)		
N1	Número de puntos introducidos	
FS	Factor de escala	
OY	Flag de orientación eje Y (OY = -1 indica origen arriba a la izquierda)	
S(2, 20)	Flag izquierda/derecha en el análisis horizontal	
T(2, 20)	Flag arriba/abajo en el análisis vertical	

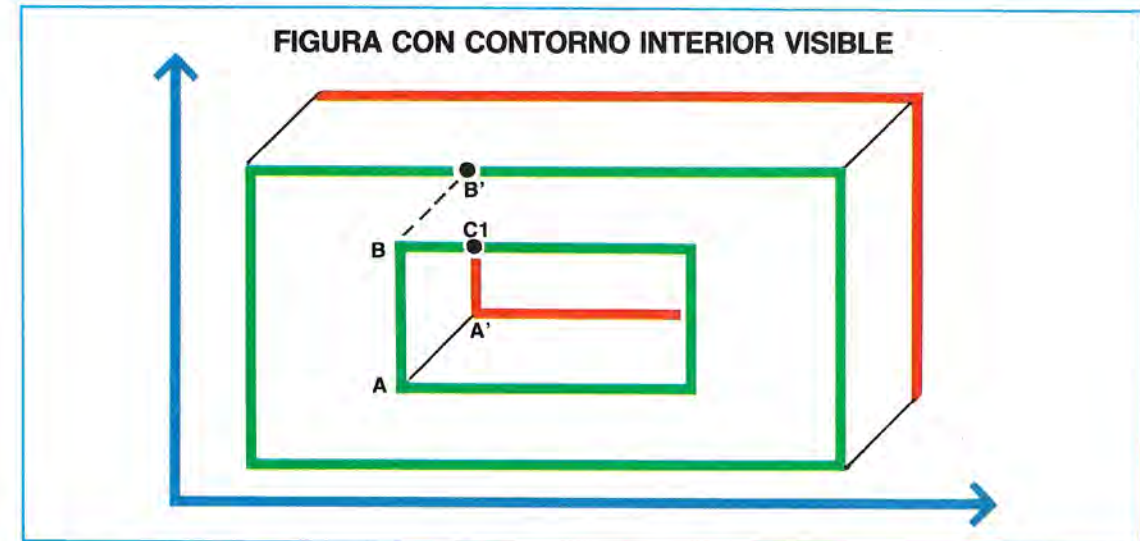
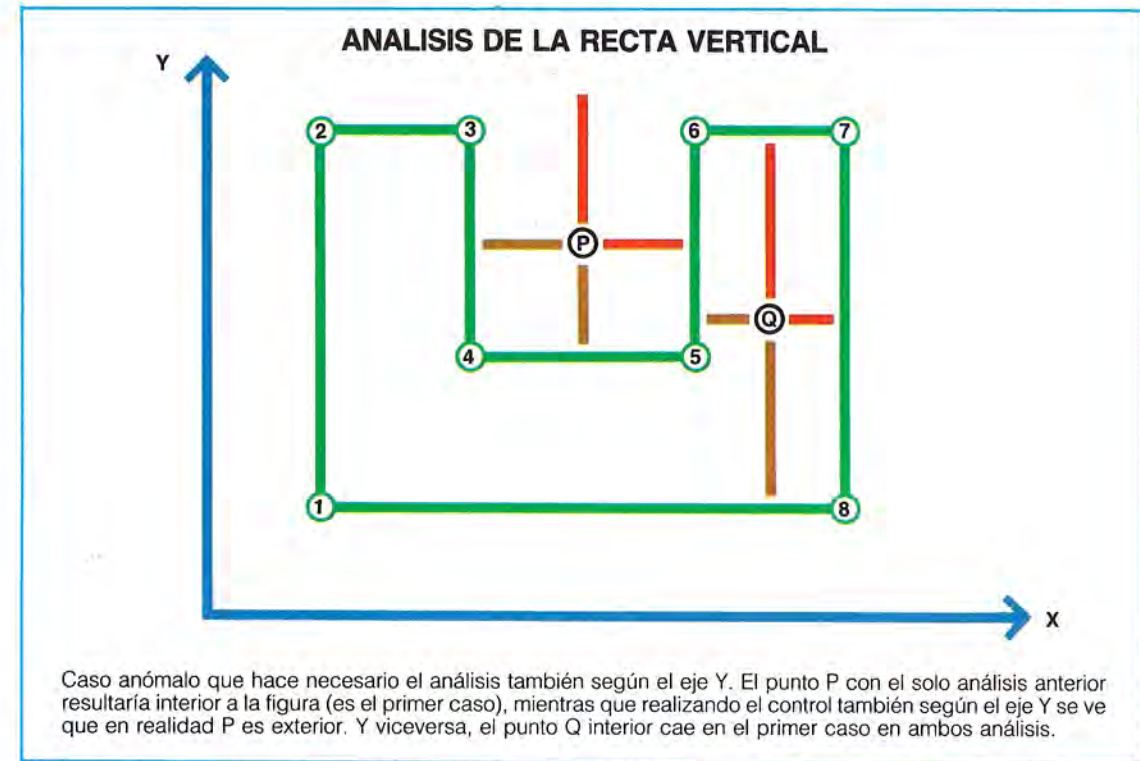
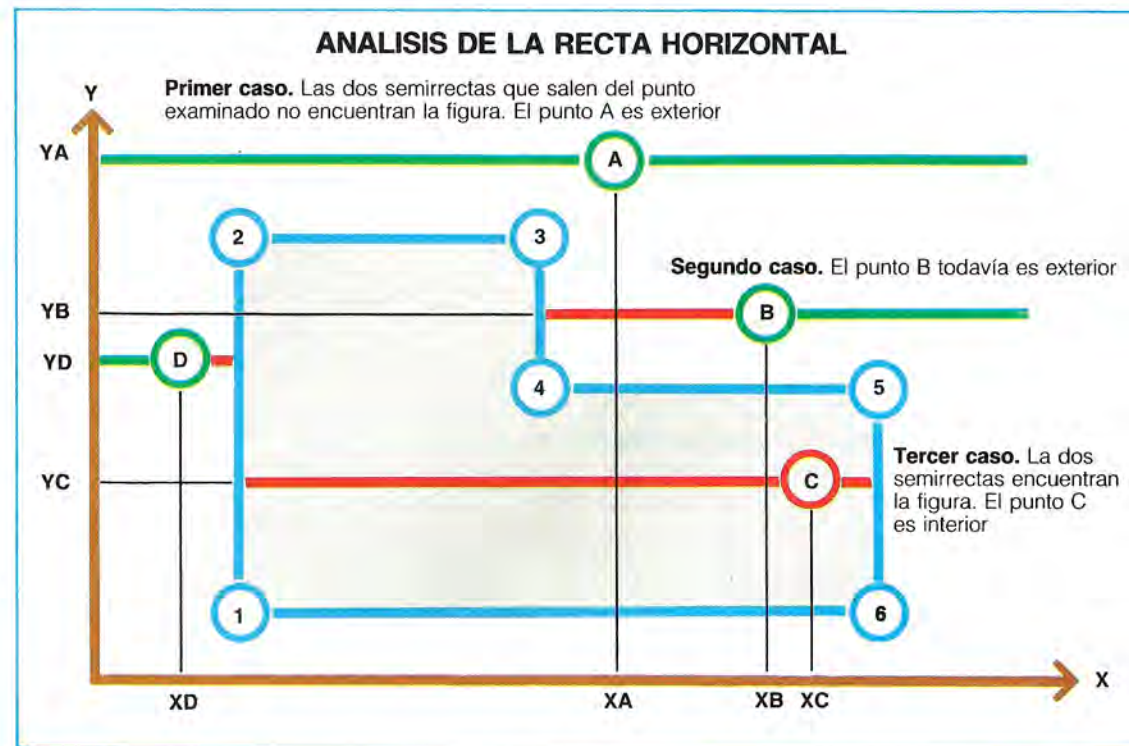
■ Determinación del tipo de cada punto de la figura trasladada. Obtenidas las coordenadas de los vértices debe comprobarse si el segmento que une dos a dos los puntos correspondientes es visible o no. En la figura de la página 1642, por ejemplo, el vértice trasladado 1 está unido al 2 por una línea que no es visible; y viceversa, el segmento que une los puntos trasladados 2 y 3 debe presentarse. La subrutina 3000 analiza cada par de vértices y determina si el lado que los une debe presentarse o no.

El método utilizado permite determinar cuándo un punto del contorno trasladado cae en el interior del contorno de base. Por ejemplo, el punto trasladado 1 está definido como no visible, puesto que es interior a la figura de base y queda cubierto por ésta. La primera limitación se debe precisamente a este método; si el espesor, y por tanto el desplazamiento D, es tan grande que genera coordenadas del punto trasladado fuera del contorno de la base, también el punto trasladado 1 queda visible, con resultados del todo erróneos. Abajo se ha mostrado el algoritmo utilizado para determinar cuándo un punto general es interno a un determinado contorno. El método se basa

en la hipótesis de que las coordenadas de los vértices se hayan introducido de manera secuencial. En la figura, la numeración de 1 a 6 indica los vértices del contorno de base, mientras que las letras de A a D indican los puntos de los cuales se quiere conocer la posición con respecto a la figura. Un punto general es interno a la figura cuando, trazando de éste dos semirrectas horizontales, cada una de ellas encuentra el contorno de la figura.

En la figura de arriba de la página siguiente se muestra un tipo particular de contorno para el cual no es suficiente analizar únicamente la recta horizontal. Para el punto P se tendrían efectivamente dos intersecciones con la figura de base, a pesar de que P es exterior. Por tanto, el análisis debe realizarse también según el eje Y. Una vez determinado el tipo de cada punto (interno/externo) puede trazarse el contorno trasladado (subrutina 5000), y por tanto pueden unirse los vértices con los segmentos que simulan el espesor (subrutina 6000).

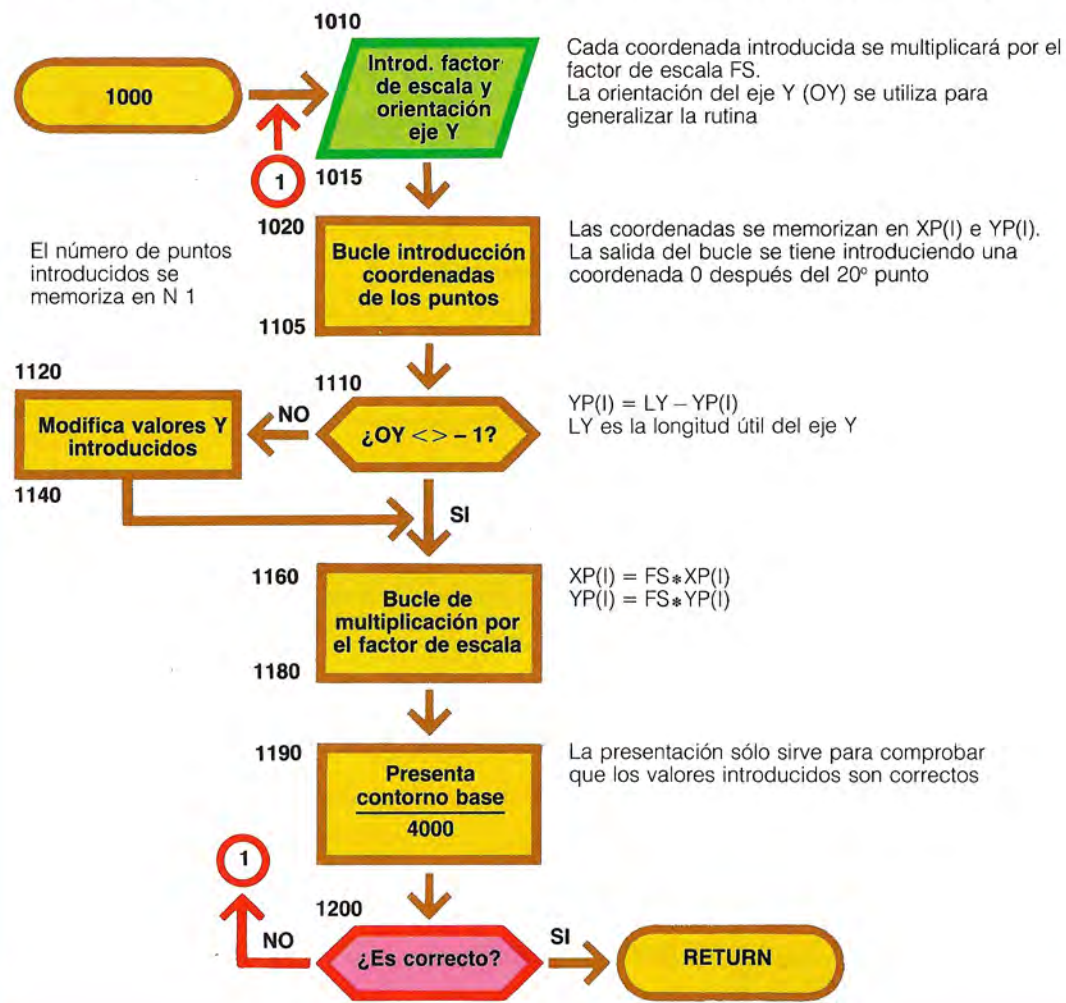
Dados los supuestos con los que se acaba de trazar un dibujo, para descartar un lado basta con que uno de los dos extremos sea interno. Con este método, el lado que une los vértices trasladados 1 y 2 no quedaría visible. Sin embargo, existen casos en los que la lógica descri-



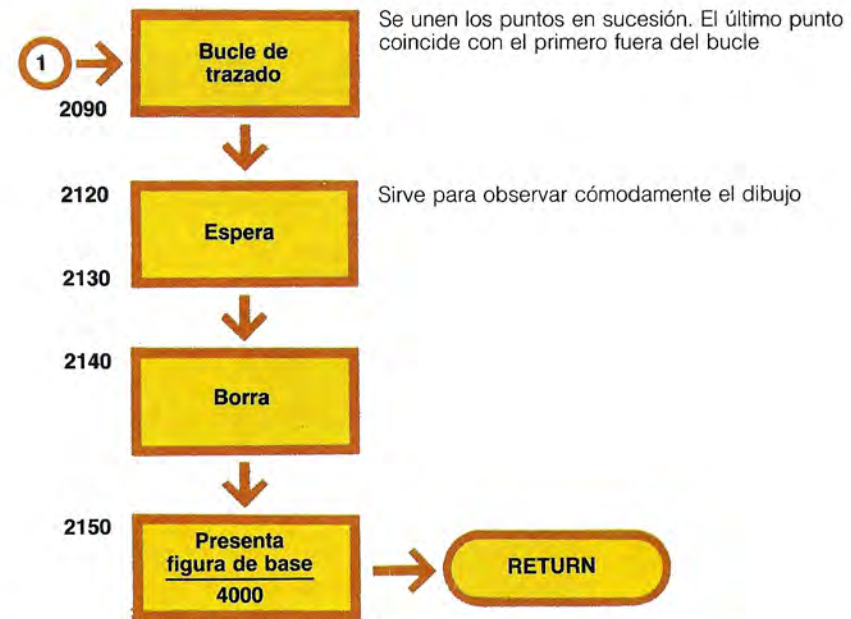
ta produciría la omisión de partes enteras del dibujo: por ejemplo, con el contorno de la figura de arriba de esta página, la parte interna quedaría completamente ignorada. Por este motivo se ha previsto la posibilidad de determinar cuándo existen partes de lado visibles y cuáles son (subrutina 5500). Introduciendo también esta implantación, determinadas zonas del dibujo pueden resultar erróneas (segmentos 2-a y 6-b de la figura de la página 1642). Este error se

debe a la selección de los puntos realizada, que se limita a analizar su posición sólo con respecto a la figura de base. La subrutina 5500 se ha incluido también para mostrar un método muy utilizado en el trazado de gráficos para determinar qué puntos todavía están incluidos en la ventana que se utiliza. En las páginas 1646 a 1657 se han representado los diagramas de flujo y el listado de todo el procedimiento.

INTRODUCCION TABLA DESPLAZAMIENTOS DE LA FIGURA DE BASE

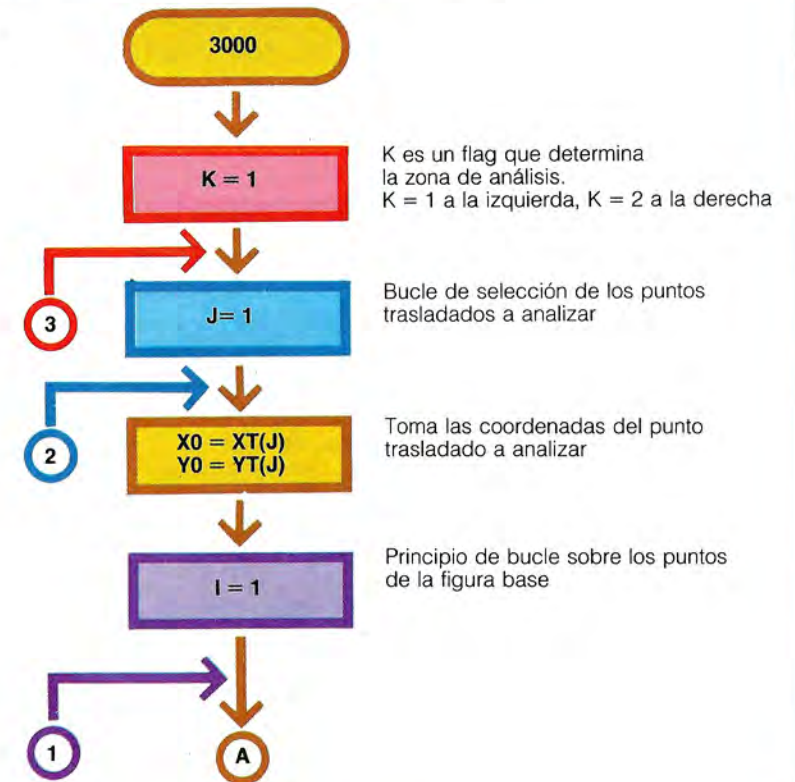


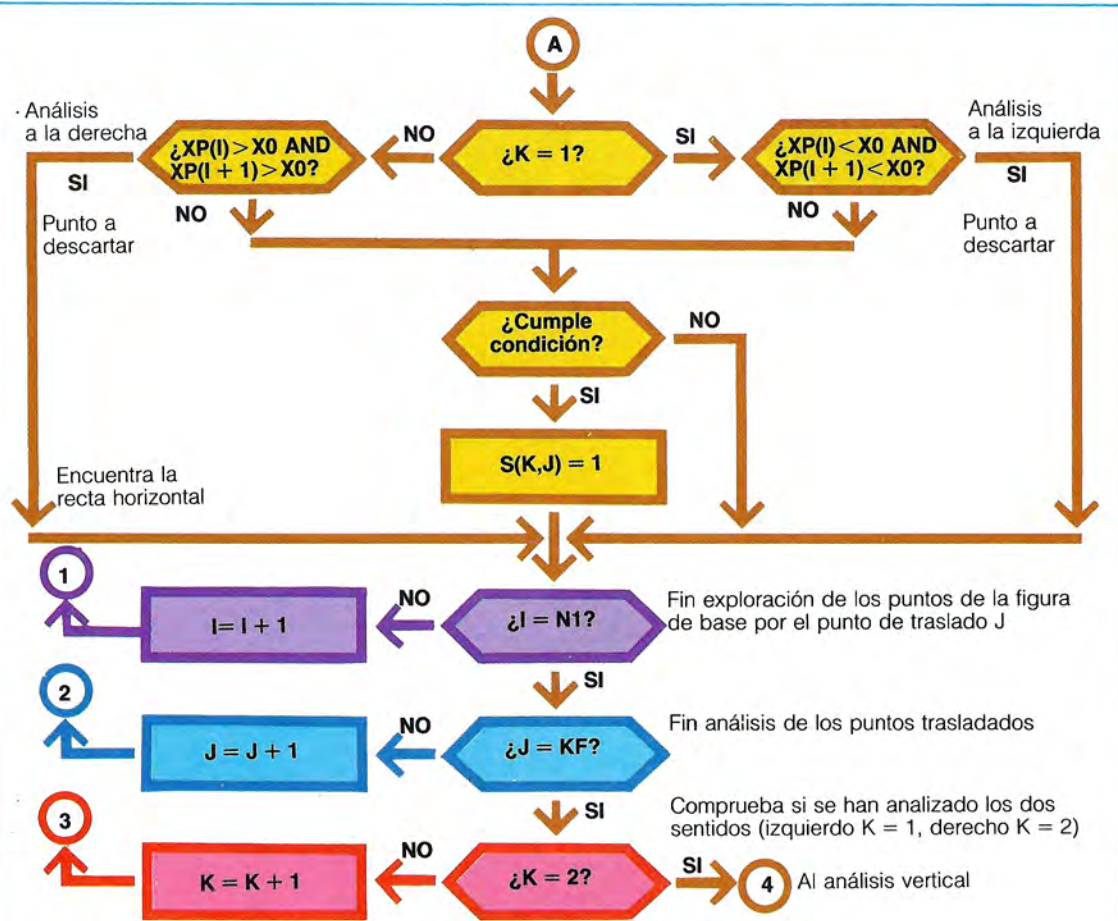
CALCULO COORDENADAS PUNTOS TRASLADADOS



DETERMINACION DEL TIPO DE PUNTO

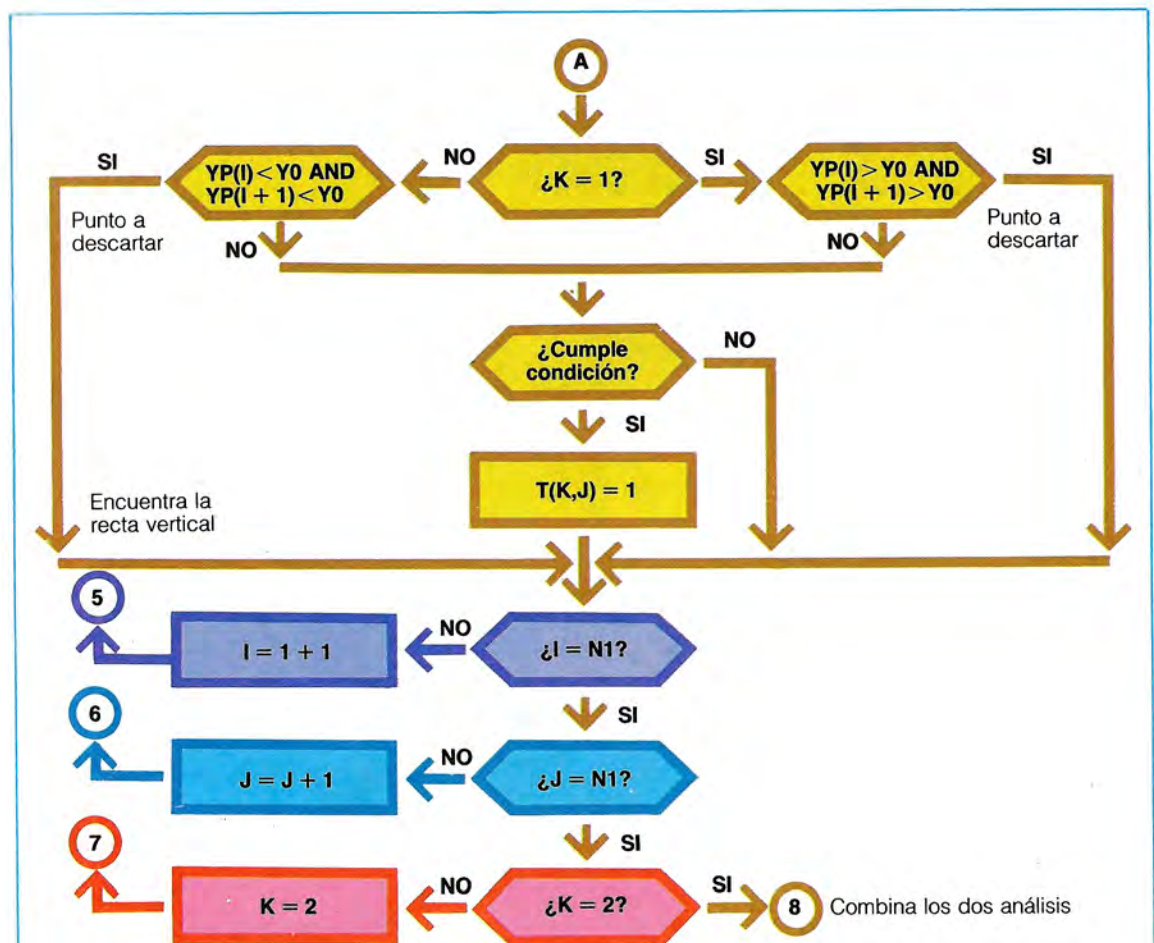
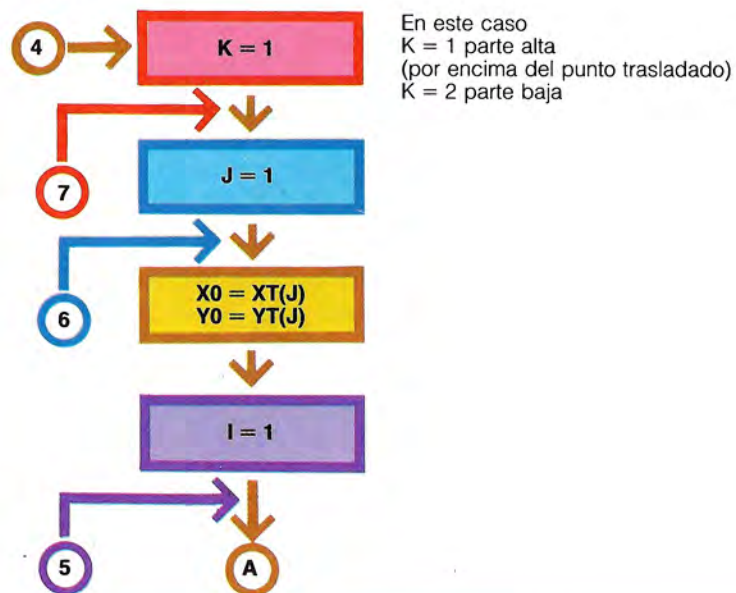
Análisis a la derecha



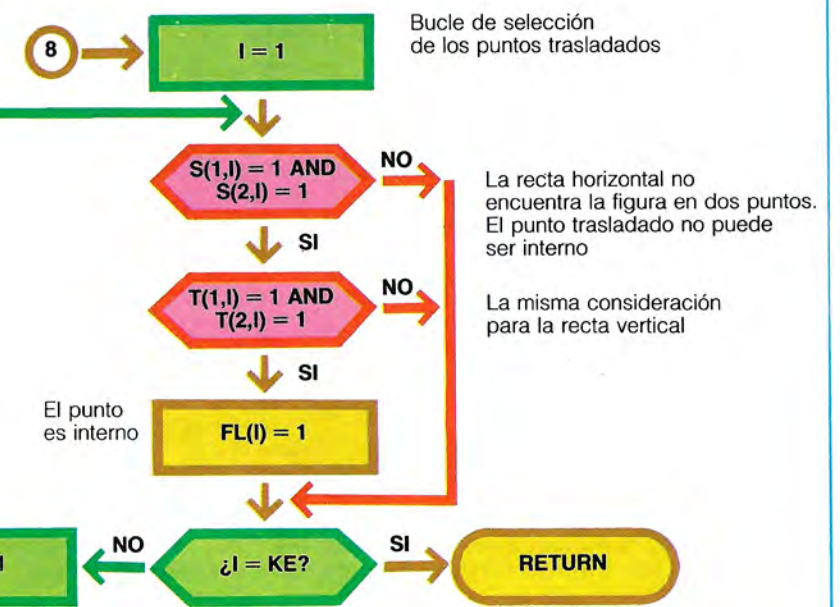


El diagrama de flujo está simplificado con respecto al listado. Algunos controles se han omitido

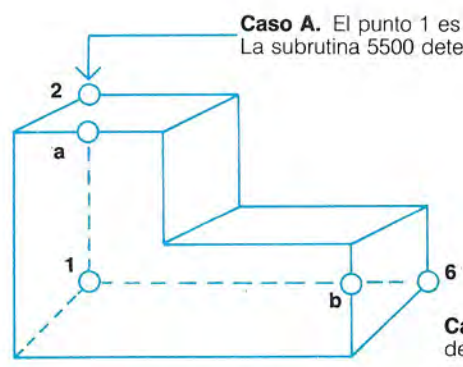
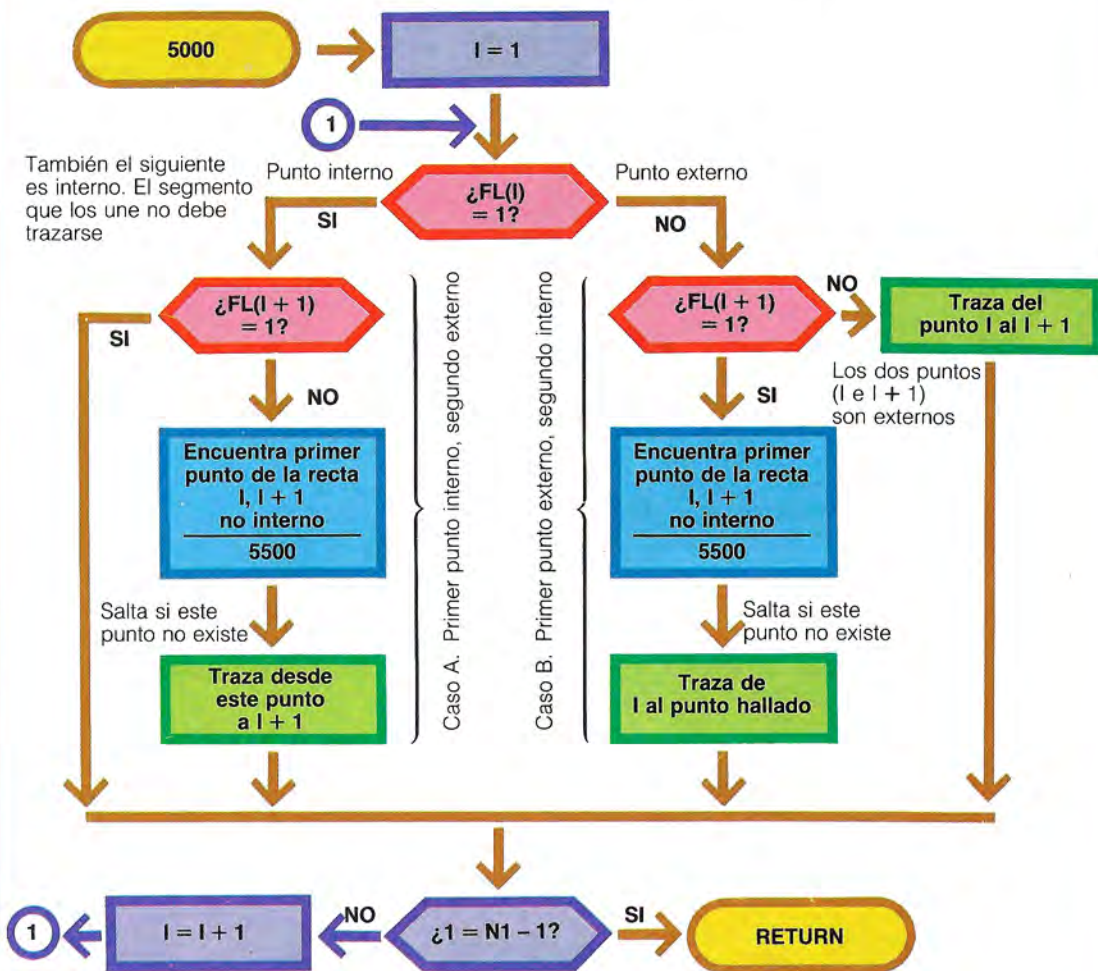
Análisis sobre la recta vertical



Análisis final

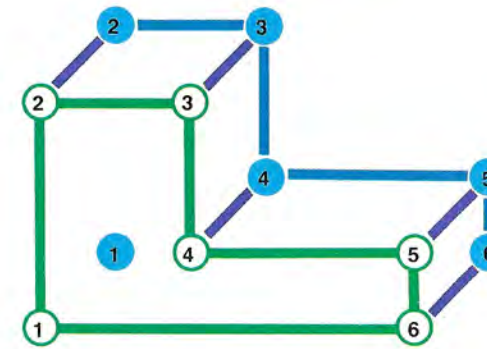
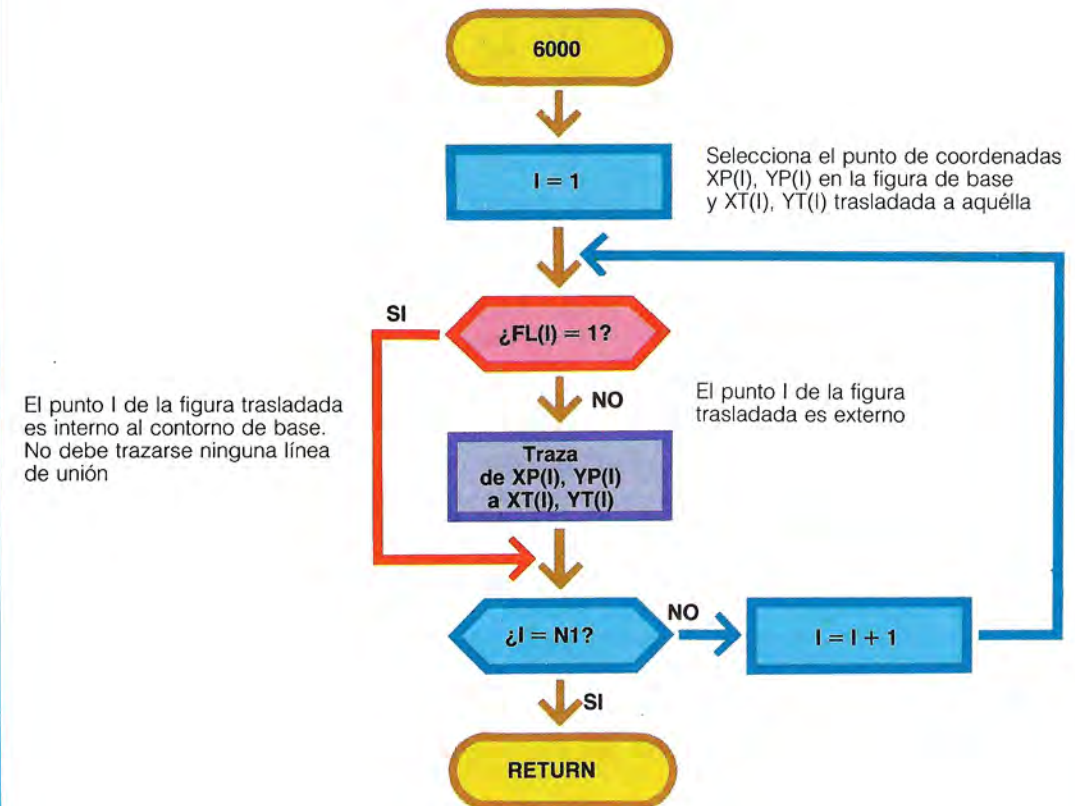


PRESENTACION FIGURA TRASLADADA



El método presentado analiza la posición de los puntos trasladados sólo con respecto a la figura de base y traza algunos segmentos erróneos, como por ejemplo los trazos a, 2 y 6, b. Para tener una versión completa debe analizarse cada punto trasladado también con respecto a las superficies que se generan para la traslación, o sea según el espesor.

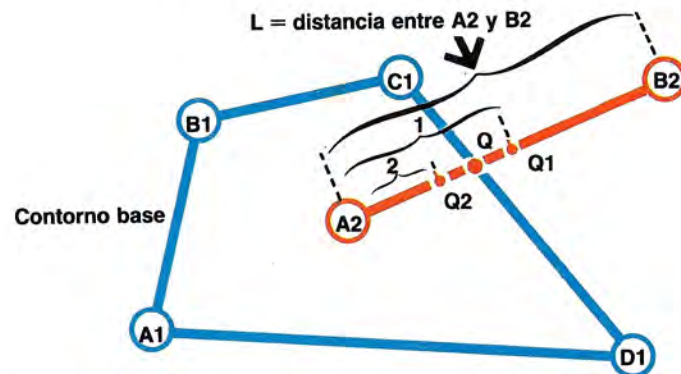
LINEAS DE UNION



- Figura de base
- Puntos trasladados [XT(I), YT(I)]
- Contorno de la figura trasladada, presentado por la 5000 (menos la 5500)
- Segmentos de unión trazados por esta rutina

Los dos puntos ① no están unidos porque el que pertenece a la figura trasladada es interno al contorno de base.

ALGORITMO UTILIZADO EN LA 5500



A1 B1 C1 D1 son el contorno de base; A2 y B2 son dos puntos generales del contorno trasladado, el uno interno y el otro externo

Se debe determinar el punto Q en el segmento A2,B2, a partir del cual se sale del contorno de base

El segmento L que une los dos puntos A2 y B2 está dividido por la mitad, definiendo así el punto Q1

Pueden producirse dos casos

1. Q1 es externo: el análisis prosigue considerando la parte A2,Q1
2. Q1 es interno: el análisis prosigue considerando la parte Q1,B2

Seleccionando el segmento sobre el que proseguir, el programa vuelve al primer paso sustituyendo los dos extremos del segmento (A2,B2) con los definidos en los pasos siguientes

El bucle se repite cuando la distancia entre los dos puntos Q sucesivos (Qn y Qn + 1) es inferior a un valor prefijado (precisión que se quiere obtener)

DETERMINACION DEL PUNTO DE TRANSICION INTERNO/EXTERNO

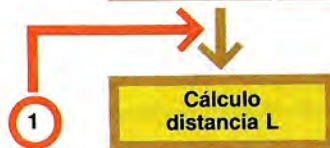
$$XA = XT(l) \quad XB = XT(l + 1)$$

$$YA = YT(l) \quad YB = YT(l + 1)$$



$$Y = M * X + N$$

En la llamada, las variables XA,YA y XB,YB deben contener las coordenadas de los extremos del segmento a analizar

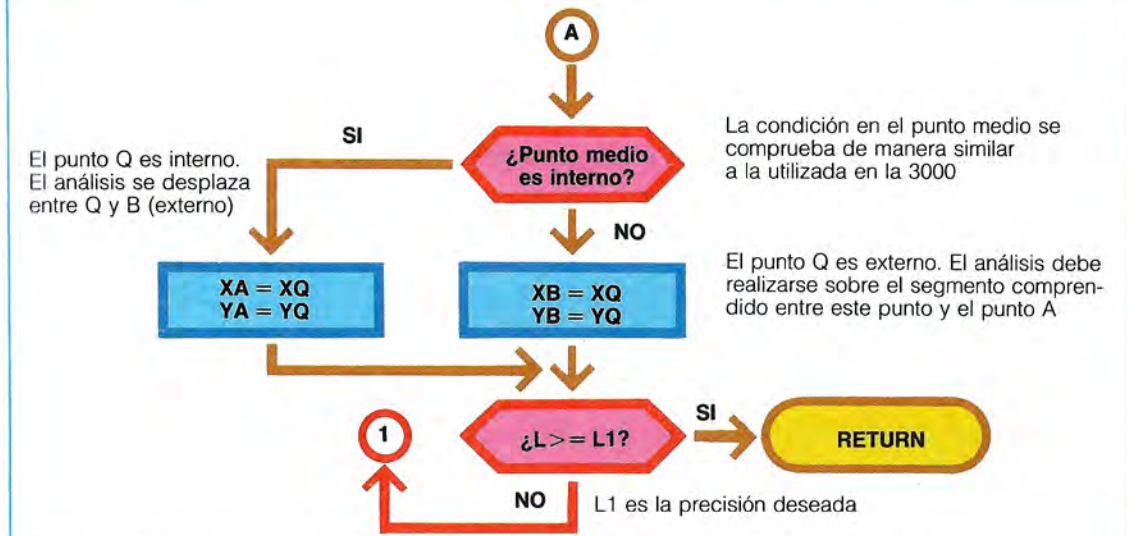


$$L = \text{SQR}((XA - XB)^2 + (YA - YB)^2)$$



$$XQ = (XA + XB)/2$$

$$YQ = M * XQ + N$$



El método ilustrado permite soluciones aproximadas. Se interrumpe la búsqueda cuando se obtiene la precisión deseada (L1), que puede no ser suficiente desde el punto de vista gráfico. Un resultado mejor se puede obtener con otro sistema que se base en la lectura de la memoria vídeo. Básicamente se trata de moverse a lo largo del segmento A2,B2 verificando, para cada punto, el estado del vídeo. Las coordenadas de Q son aquellas en que el pixel es activo. El método permite resultados mejores desde el punto de vista gráfico, pero emplea más tiempo, teniendo que analizar el contenido de todas las memorias correspondientes al segmento A2,B2. Uniendo los dos sistemas puede obtenerse una optimización. Con el primero se determina el entorno en el que cae Q, después se analiza este entorno comprobando el contenido de la memoria vídeo

La subrutina 5500 puede sustituirse por la 5500 XQ =XB:YQ = YB:RETURN

De esta manera se reducen las posibilidades de aplicación del programa, pero se evita el error

GRAFICO TRIDIMENSIONAL CON BORRADO DE LAS LINEAS OCULTAS

```

1  REM =====
2  REM *PROGRAMA PARA*
3  REM *FIGURAS EN 3D*
4  REM * Y BORRADO *
5  REM *DE LAS LINEAS*
6  REM * OCULTAS *
7  REM =====
8  :
9  :
50 S = 20:L1 = 5:LY = 191
55 DIM YP(20),XP(20),YT(20),XT(20)
    0),FL(20),S(2,20),T(2,20)
60 ONERR GOTO 10000
65 CN = 1
100 GOSUB 1000
110 GOSUB 2000
120 GOSUB 3000
130 GOSUB 4000
140 GOSUB 5000
150 GOSUB 6000
200 HOME : VTAB 21: INPUT "OTRO
    GRAFICO ? ":RS#
210 IF LEFT$(RS#.1) = "S" THEN
    RUN
    
```

```

220 TEXT: HOME : VTAB 22: END
999 :
1000 REM =====
1001 REM *INT/FIG/BASE*
1002 HGR
1003 :
1005 TEXT

1010 HOME : VTAB 10: INPUT "INSER
TAR FACTOR DE ESCALA ":FE
1015 VTAB 10: HTAB 10: INPUT "ORIE
NACION ORDENADA ":OY
1020 I = 1
1030 VTAB 10: HTAB 10: PRINT "CO
ORDENADAS PUNTO "I" "
1040 VTAB 10: HTAB 31: INPUT "X=
":XP(I)
1050 IF XP(I) * FS > 278 THEN 10
40
1060 VTAB 12:HTAB 31: INPUT "Y=
":YP(I)
1070 IF YP(I) * FS > 191 THEN 10
60
1080 IF YP(I) = 0 OR XP(I) = 0 THEN
1100
1090 IF I = 20 THEN 1100
1095 I = I + 1: GOTO 1030
1100 N1 = I - 1
1105 XP(N1 + 1) = XP(N1):YP(N1 +
1) = YP(N1)
1110 IF OY < > - 1 THEN 1160
1120 FOR I = 1 TO N1
1130 YP(I) = LY - YP(I)
1140 NEXT I
1160 FOR I = 1 TO N:
1170 XP(I) = FS * XP(I):YP(I) = F
S * YP(I)
1180 NEXT I
1190 GOSUB 4000
1200 VTAB 22: INPUT "DIBUJO CORR
ECTO? ":RS#
1210 IF LEFT$(RS#.1) = "S" THEN
RETURN
1220 GOTO 1000
2000 :
2001 REM =====
2002 REM * CALCULOS DE LOS *
2003 REM *PUNTOS TRASLADADOS*
2004 REM =====
2005 :
2010 FOR I = 1 TO N1
2020 XT(I) = S + XP(I):YT(I) = YP
(I) - S
2030 NEXT I
2035 XT(I) = XT(N1) + S:YT(I) = Y
T(N1) - S
2040 HOME : VTAB 22: INPUT "PRES
ENTO FIGURA TRASLADADA? ":RS#
2050 IF LEFT$(RS#.1) = "S" THEN
2070
2060 RETURN
2070 :
2090 FOR I = 1 TO N1 - 1

```

```

2100 H PLOT XT(I).YT(I) TO XT(I+
1).YT (I + 1)
2120 NEXT I
2120 H PLOT XT (N1 - 1).YT(N1 - 1)
TO XT(N1).XT(N1)
2130 VTAB 22: PRINT "PULSAR UNA
TECLA PARA BORRAR ": GET
Q#
2140 HGR
2150 GOSUB 4000
2160 RETURN
2999 :
3000 REM =====
3001 REM * DETERMINACION*
3002 REM *TIPO DEL PUNTO*
3003 REM =====
3004 :
3005 KE = N1: HOME: VTAB 22: HTAB
10: PRINT "ESPERAR"
3006 REM -----
3007 REM :ANALISIS RECTA:
3008 REM : HORIZONTAL :
3009 REM -----
3010 FOR K = 1 TO 2
3020 FOR J = 1 TO KE
3025 IF KE = THEN 3040
3030 X0 = XT(J):Y0 = YT(J)
3040 FOR I = 1 TO N1
3050 IF K = 1 THEN 3070
3060 IF XP(I) > = X0 AND XP (I +
1) > = X0 THEN 3100
3065 GOTO 3080
3070 IF XP (I) = < X0 AND XP (I +
1) < = X0 THEN 3100
3080 IF (YP(I) < = Y0 AND YP (I +
1) > = Y0) OR (YP(I) > = Y
0 AND YP(I + 1) < = Y0) THEN
S(K,J) = 1
3100 NEXT I.J.K
3105 :
3106 REM -----
3107 REM :ANALISIS RECTA:
3108 REM : VERTICAL
3109 REM -----
3110 FOR K = 1 TO 2
3120 FOR J = 1 TO KE
3125 IF KE = 1 THEN 3140
3130 X0 = XT(J):Y0=YT(J)
3140 FOR I = 1 TO N1
3150 IF K = 1 THEN 3170
3160 IF YP(I) > = Y0 AND YP(I +
1) > = Y0 THEN 3200
3165 GOTO 3180
3170 IF YP(I) < = Y0 AND YP (I +
1) < = Y0 THEN 3200
3180 IF (XP(I) < = X0 AND XP(I +
1) > = X0) OR (XP(I) > = X
0 AND XP(I + 1) = < X0) THEN
T(K,J) = 1
3200 NEXT I.J.K.
3205 :
3206 REM -----
3207 REM :ANALISIS FINAL:
3208 REM -----
3210 FOR I = 1 TO KE
3220 IF S(1.I) = 1 AND S(2.I) =
1 THEN 3240

```



```

3230 GOTO 3250
3240 IF T(1,I) = 1 AND T(2,I) =
      1 THEN FL(I) = 1
3250 NEXT I
3260 RETURN
3999 :
4000 REM =====
4001 REM * REPRESENTA *
4002 REM * FIGURA BASE *
4003 REM =====
4004 :
4005 HGR : HCOLOR= 3
4010 FOR I = 1 TO N1 - 1
4020 HPLOT XP(I),YP(I) TO XP(I +
      1),YP(I + 1)
4030 NEXT I
4035 GOTO 4060
4040 HPLOT XP(N1 - 1),YP(N1 - 1)
      TO XP(N1),YP(N1)
4060 RETURN
4999 :
5000 REM =====
5001 REM *PRESENTA FIG.*
5002 REM * TRASLADADA *
5003 REM =====
5004 :
5010 FOR P = 1 TO N1 - 1
5020 IF FL(P) = CN THEN 5200
5030 IF FL(P + 1) = CN THEN 5050

5040 HPLOT XT(P),YT(P) TO XT (P+
      1),YT(P + 1)
5045 GOTO 5300
5050 XB = XT(P):XA = XT(P + 1):YB
      = YT(P):YA = YT(P + 1)
5055 GOSUB 5500
5060 HPLOT XT(P),YT(P) TO XQ,YQ
5070 GOTO 5300
5100 :
5200 IF FL(P + 1) = CN THEN 5300

5210 XB = XT(P):XA = XT(P + 1)
5211 YA = YT(P):YB = YT(P + 1)
5215 GOSUB 5500
5220 HPLOT XQ,YQ TO XT(P + 1),YT
      (P + 1)
5300 NEXT P
5310 RETURN
5499 :
5500 REM =====
5501 REM * PUNTOS DE TRANSF. *
5502 REM * INTERIOR/EXTERIOR *
5503 REM *=====
5504 :
5505 IF XA = XB THEN X = 1: GOTO
      5515
5506 REM -----
5507 REM :YQ = XQ * M + N:
5508 REM -----
5510 X = XB - XA
5515 M = (YB - YA) / X:N = INT (
      YA - (XA * M))
5520 L = INT ( SQR ((XA - XB) ^
      2 + (YA - YB) ^ 2))

```

```

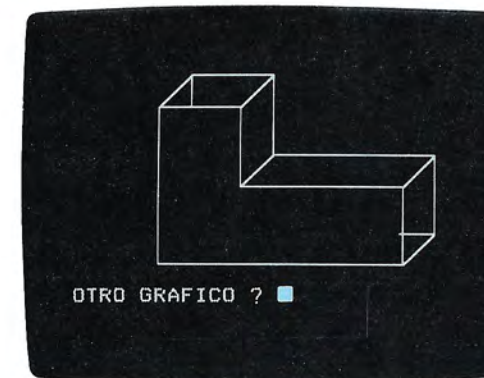
5525 IF X = 1 THEN XQ = XA: YQ =
      INT, ((YA + YB) / 2): GOTO 5
      540
5530 XQ = INT ((XA + XB) / 2):YQ
      = INT (M * XQ + N)
5540 X0 = XQ:Y0 = YQ
5545 S(1,1) = 0:S(2,1) = 0:T(1,1)
      = 0:T(2,1) = 0
5550 CF = FL(1):KE = 1:FL(1) = 0:
      GOSUB 3010
5580 IF FL(1) = ON THEN XA = XQ:
      YA = YQ: GOTO 5600
5590 XB = XQ:YB = YQ
5600 FL(1) = CF
5605 IF L <= L1 THEN SW = 0: RETURN

5610 GOTO 5520

5999:
6000 REM =====
6001 REM * LINEAS DE *
6002 REM * CONJUNCION *
6003 REM =====
6004 :
6010 FOR I = 1 TO N1
6020 IF FL(I) = CN THEN 6040
6030 HPLOT XP(I),YP(I) TO XT(I),
      YT(I)
6040 NEXT I
6050 RETURN
10000 :
10001 REM =====
10002 REM * ERROR *
10003 REM =====
10004 :
10005 TEXT : HOME
10010 ER = PEEK (222)
10020 IF ER = 53 THEN VTAB 10: PRINT
      "COORDENADAS ERRONEAS ": GOTO1
      0050
10030 VTAB 10: PRINT "DATOS ERRO
      NEOS "
10050 VTAB 22: HTAB 30: PRINT ">
      >":
10060 GET Q$: RUN

```

Un ejemplo de funcionamiento del programa presentado. A la izquierda, el programa ha analizado los puntos de la figura base y ha representado, trasladados, los que se ven. A la derecha, el dibujo se ha completado. Obsérvense los dos segmentos de error.



La animación de imágenes en el ordenador

Una clase particular de aplicaciones del gráfico de ordenadores es la que corresponde a la animación de las figuras gráficas. Las técnicas de animación se utilizan ampliamente en los programas de videojuegos, pero su aplicación no siempre es de tipo tan frívolo. Existen numerosísimos ejemplos en los que la animación de una figura en el monitor puede simplificar enormemente una fase de salida numérica, por ejemplo cuando se quiere ver el efecto de una fuerza sobre una estructura sólida u observar el movimiento de un órgano mecánico.

El ejemplo más sencillo de estructura orientada a la gestión de la animación de figuras gráficas es la gestión de los sprites.

Los sprites

Hemos visto que la presentación de una imagen gráfica no es una operación elemental; por el contrario, requiere el trazado de un elevado número de segmentos, tantos como sirven para completar la imagen. El desplazamiento de la imagen así obtenida requiere su borrado y una nueva fase de trazado en otra posición.

En cambio, el **sprite** (duendecillo) es una imagen gráfica que puede gestionarse como tal desde el programa, eligiendo de una vez por todas la forma y, de vez en cuando, el punto de la pantalla en que debe presentarse.

El sprite, que generalmente tiene dimensiones limitadas con respecto a la pantalla de vídeo, posee tres características principales:

- es completamente definible por el usuario
- puede moverse en las cuatro direcciones del plano
- permite detectar eventuales colisiones con otros sprites y con las otras figuras presentes en el monitor.

Cada una de estas características puede implantarse con subrutinas dedicadas para simular los sprites también en los sistemas en que no están previstos.

Las máquinas que prevén los sprites también poseen oportunas instrucciones que sirven para moverlos, para verificar eventuales colisiones, etc. Por tanto, están dotados de algunas implantaciones no previstas en el Basic estándar y, como tales, diferentes en cada máquina. General-

mente, la gestión de los sprites está controlada por componentes hardware dedicados; para las máquinas que no poseen estos componentes es necesario construir rutinas adecuadas que simulan el hardware que falta. En este caso se tendrán algunas limitaciones en las funciones que pueden realizarse.

Gestión software de los sprites

El uso de los sprites está estrechamente ligado a la producción de dibujos animados, y precisamente a este uso se deben algunas de las dificultades en la gestión vía software.

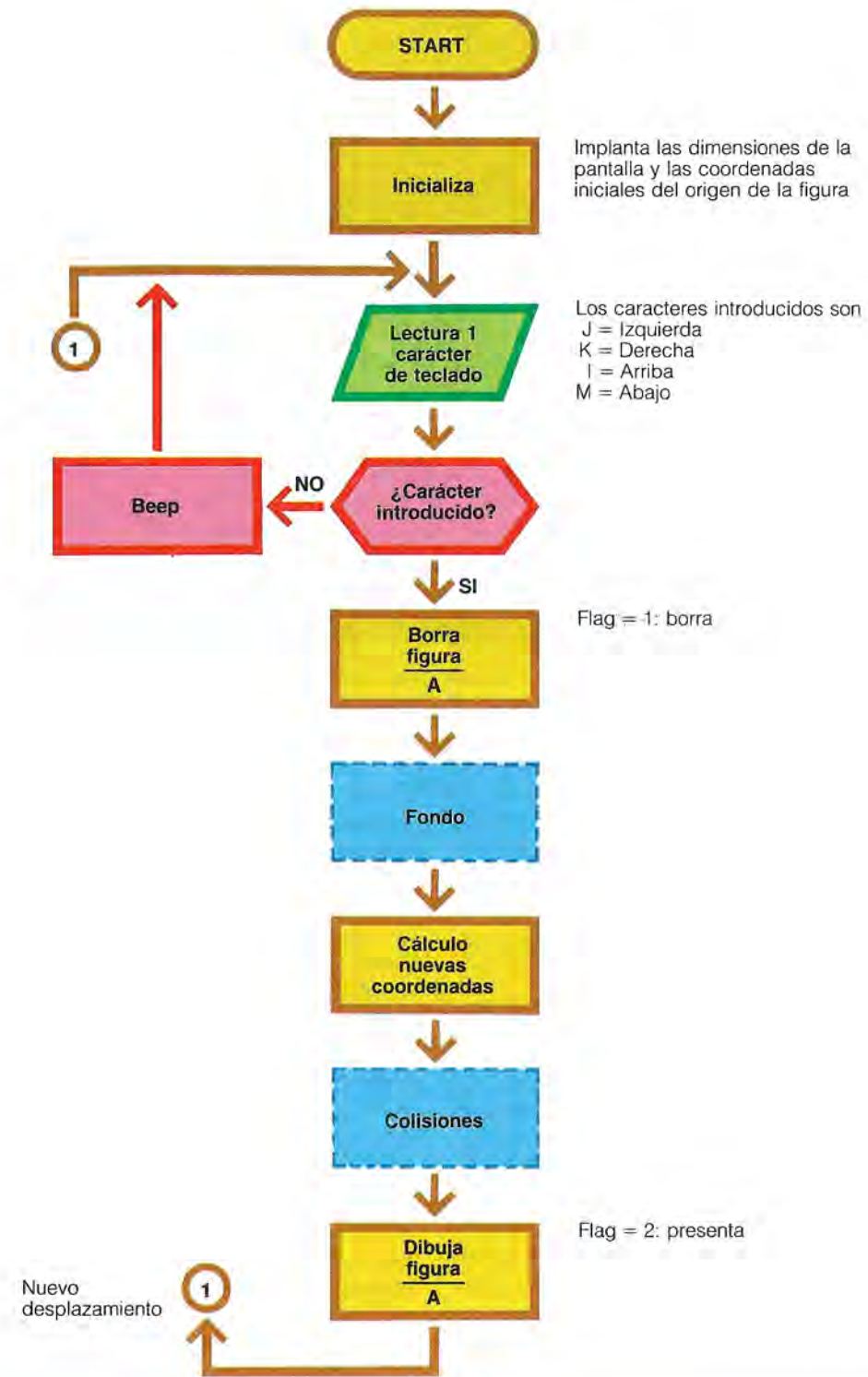
Un sprite puede crearse como tabla de las figuras y desplazarse simplemente variando el origen. Esta técnica, que ya existe en diversas aplicaciones, consiste en generar la figura a mover (sprite) como una serie de desplazamientos desde un origen.

Variando las coordenadas del punto elegido como referencia se tiene el desplazamiento de la figura (naturalmente debe estar previsto el borrado en la zona anterior al desplazamiento).

En la página siguiente se ha representado el diagrama de flujo de un programa que desplaza una figura en el vídeo siguiendo los comandos impartidos por teclado. Las funciones realizadas se ven inmediatamente con el examen del diagrama de flujo; la única observación corresponde a la rutina indicada con el símbolo A, que realiza las dos funciones de borrado y presentación, y que debe ser llamada implantando oportunamente un flag para seleccionar la deseada. En el diagrama de flujo se han indicado además dos bloques (Fondo y Colisiones), cuyo empleo se especifica en esta aplicación particular.

La animación de imágenes gráficas. La animación con el ordenador es una técnica muy similar a la utilizada en los empleos cinematográficos. La escena debe contener algunas figuras fijas (el fondo), con respecto a las cuales debe moverse el sujeto de la animación. En cinematografía, la técnica consiste en dibujar una tabla sólo con el fondo y tantas tablas, sobre un soporte transparente, como cuantos sean los movimientos que se desea obtener. En cada una de estas tablas se reproduce la figura que debe desplazarse, en posición ligeramente modificada con respecto a la anterior. Superponiendo estas tablas sobre el fondo y fotografiando cada vez, se obtiene una película que, proyectada, da la sensación de movimiento.

DESPLAZAMIENTO DE UNA FIGURA





P.A. Simon/Grazia Neri-Phototake

Un momento del proceso con el ordenador de las imágenes del filme "Tron".

En realidad, el problema no es tan sencillo como podría parecer. La figura que debe moverse puede gestionarse de dos maneras:

- 1 / La figura, durante el desplazamiento relativo al fondo, no varía; por tanto, sólo se trata de realizar un posicionado diferente
- 2 / La figura, además de moverse con respecto al fondo, sufre modificaciones.
En este caso, además del desplazamiento, para cada posición debe volverse a dibujar la figura, toda o en parte

El segundo procedimiento, muy sencillo de realizar con papel y lápiz, presenta algunas dificultades para ser realizado con el ordenador. La primera dificultad principal consiste en la recuperación del fondo después de un movimiento. Un cierto punto de la pantalla pertenece al fondo hasta que su zona no está interesada por la figura en movimiento. Por tanto es activo (en visión) o apagado según si debe o no verse el fondo. Cuando el mismo punto está interesado por la figura en movimiento, es controlado

por el gráfico de esta última, y después el desplazamiento debe recuperarse en la situación inicial correspondiente al fondo. Por ejemplo, si un hombre que pasea pasa por delante de un árbol, el ordenador deberá sustituir de vez en cuando la figura del hombre por una parte del árbol, para poder recuperar la imagen apenas el hombre haya pasado. Si además la figura puede sufrir modificaciones durante el desplazamiento, no se trata ya simplemente de trasladar un gráfico, sino de prever también las modificaciones.

En las máquinas que tienen los sprites, la gestión del fondo está a cargo del sistema: la figura móvil se construye como sprite y puede moverse sin interesar el fondo, precisamente como si estuviese dibujada sobre una hoja transparente. Sin embargo, esta gestión de los sprites no permite modificar su forma; es decir, es como si se realizase una sola de las tablas transparentes y se trasladara con respecto al fondo. La sensación es la de movimiento de una figura estática. Para realizar una animación más realista debe modificarse también la forma de la figura, por

ejemplo dibujando algunos detalles en posiciones diferentes. Utilizando los sprites, esto puede obtenerse de dos maneras: modificando la forma del sprite antes del movimiento o bien creando tantos sprites diferentes como cuantas sean las posiciones que se desean simular y presentándolas en sucesión. En este caso, cada una de las figuras es un sprite, y la animación consiste en la activación sucesiva de los diversos sprites; por tanto, se tiene un funcionamiento del todo análogo a la animación tradicional.

Estos mismos problemas se encuentran en la gestión de los sprites con programas escritos por el usuario, en los cuales debe preverse una rutina para la reconstrucción del fondo y otra para la verificación de eventuales colisiones. En las máquinas que prevén los sprites, esta función también se gestiona automáticamente.

Gestión del fondo. Existen varios métodos para gestionar el fondo de un dibujo animado, más o menos sofisticados, principalmente en función de dos parámetros:

- Velocidad de ejecución
- Ocupación de memoria

Para dar una buena sensación de continuidad en el desplazamiento, la animación de un dibujo necesita una elevada velocidad de ejecución. Si el programa no es suficientemente rápido, el ojo percibe las diversas fases del desplazamiento (borrado, reconstrucción del fondo, representación) con resultados poco agradables; además, se está vinculado a realizar desplazamientos relativamente grandes para no tener un movimiento excesivamente lento, lo cual es una situación inaceptable, por ejemplo en los videojuegos. El método y los ejemplos propuestos a continuación sólo tienen una finalidad ilustrativa y, además, para conservar una buena generalidad de empleo, no aprovechan particularidades del hardware de algunas máquinas, que en cambio podrían utilizarse para minimizar los problemas de lentitud y de ocupación de memoria. En cada caso es conveniente utilizar los programas en versión interpretada sólo en la fase de escritura y comprobación, mientras que la aplicación deberá compilarse de manera que se obtengan velocidades de ejecución notablemente superiores. Una alternativa válida es la de escribir algunas rutinas en Assembler.

El fondo puede considerarse como un dibujo que utiliza toda la pantalla de vídeo, por lo que el modo más sencillo de gestionarlo durante una animación consiste en registrarlo en una zona de memoria dedicada y separada del resto y reclamarlo cuando debe reconstruirse.

En particular, las funciones a realizar durante el desplazamiento de un sprite son:

- 1 / Transferencia del fondo de la memoria de apoyo al vídeo (presentación)
- 2 / Presentación del sprite en la posición inicial
- 3 / Representación del fondo (permite el borrado del sprite)
- 4 / Presentación del sprite en la siguiente posición

El movimiento del sprite se obtiene activando iterativamente en rápida sucesión las fases indicadas, hasta cubrir todo el desplazamiento con una serie de pequeños movimientos que dan la sensación de continuidad. Sin embargo, el efecto visual producido por este método no está entre los mejores. A causa de la baja velocidad de ejecución, común a casi todos los intérpretes Basic, puede advertirse muy bien la sucesión de las operaciones, permitiendo así una parte de la sensación de continuidad de los desplazamientos que debe caracterizar precisamente la animación. La forma más idónea de dar al software no es susceptible de generalizarse, puesto que implica el uso de soluciones hardware particulares.

Presentación de un sprite. Veamos ahora de cerca cómo puede resolverse el problema en una máquina cualquiera. Queremos presentar en la pantalla un pequeño rectángulo (sprite) de 24 (base) × 21 (altura) puntos de pantalla*, posicionado en el punto X0,Y0 sobre una imagen gráfica que constituye el fondo. Sin entrar en el contenido del rectángulo, cada uno de sus desplazamientos deberá ir acompañado de la reconstrucción de la parte del fondo enmascarada anteriormente. El procedimiento puede dividirse en tres fases sucesivas:

Fase 1. Antes del posicionado del sprite, la zona de la pantalla que será interesada (pequeño

* La elección de estos valores sirve para mantener el sprite las mismas dimensiones utilizadas por sistemas que prevén la gestión hardware de los sprites.

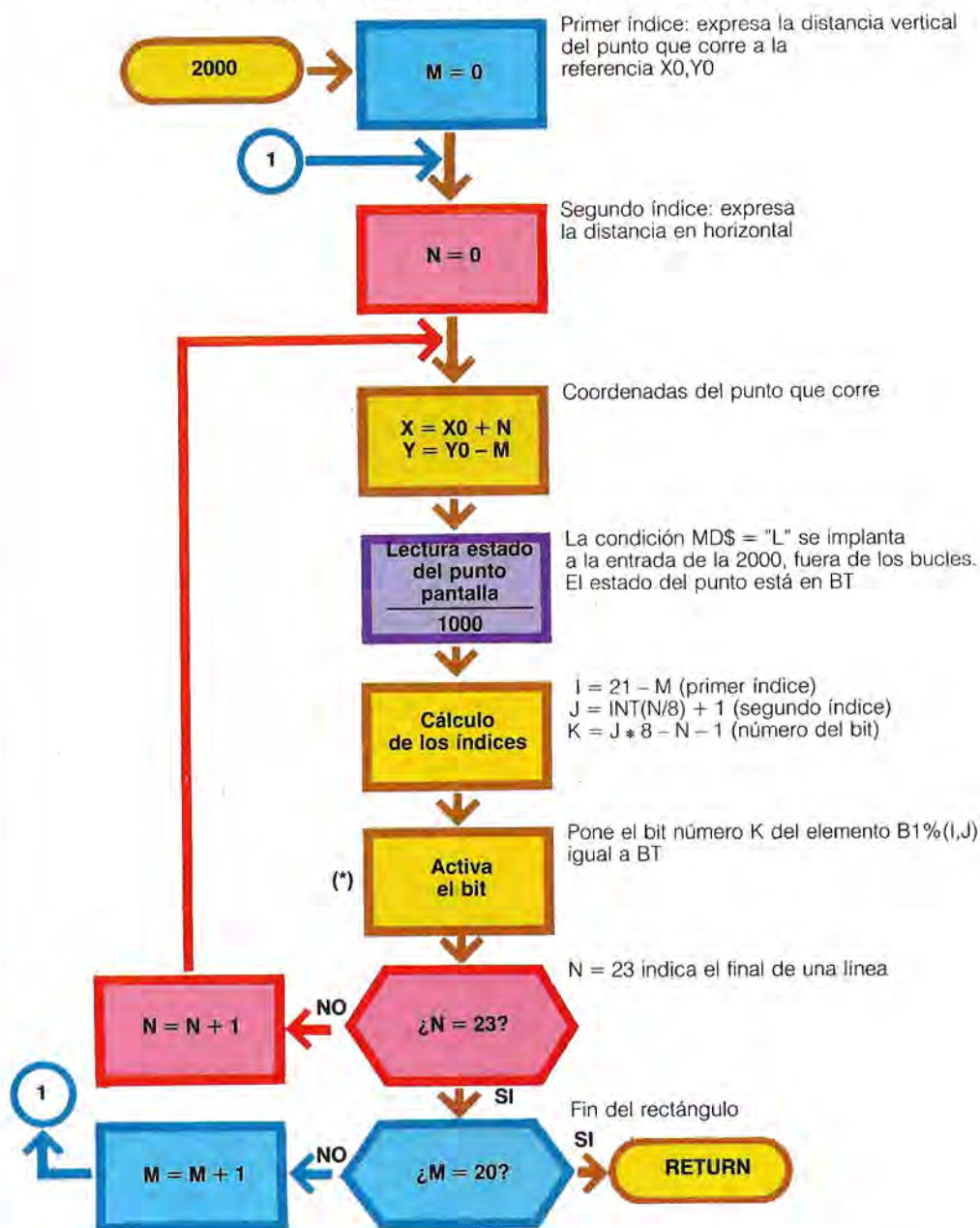
rectángulo de las mismas dimensiones del sprite) se salva en un área de memoria (matriz bidimensional). Operativamente, esta función consiste en tomar de las coordenadas de pantalla

de cada punto a salvar la correspondiente dirección de la memoria de vídeo. El contenido de esta memoria se lee y se transfiere a un buffer. En esta página se ha representado el diagrama

de flujo de la rutina; el buffer que se ha utilizado es B1%(21,3) (la explicación de este dimensionado se indica más adelante).

El listado del programa puede verse en las páginas 1667 a 1669.

MEMORIZACION DE LA VENTANA VIDEO QUE DEBERA SER OCUPADA POR EL SPRITE



* Este bloque de instrucciones es idéntico al análogo utilizado en la subrutina 1000 para activar un punto de pantalla. El método corresponde a máquinas que no permiten utilizar los operadores lógicos para crear máscaras en el interior del byte. Si esta posibilidad existe (por ejemplo bajo CP/M) basta con utilizar el oportuno operador lógico.

Fase 2. A partir del punto X0,Y0 se inserta el sprite; el contenido de la memoria vídeo correspondiente se sustituye por el contenido de la matriz S%(21,3) que representa, en el formato oportuno, el sprite.

Fase 3. El desplazamiento del sprite se obtiene recuperando el fondo, es decir transfiriendo B1%(21,3) al área de memoria vídeo ocupada por el sprite y empezando nuevamente por la fase 1 con diversos valores de X0,Y0. Las fases 2 y 3 pueden describirse con el mismo diagrama de flujo. La correspondiente rutina, parametrizada, presenta a partir de las coordenadas X0,Y0, el sprite S%(21,3) o el fondo B1%(21,3).

La transferencia de datos de y hacia la memoria vídeo requiere el cálculo de la posición de memoria correspondiente a las coordenadas en puntos de pantalla y del bit en el interior de ella. Para esto se ha adaptado la rutina 100 ya presentada, eliminando los coloquios utilizados en el ejemplo.

Esta rutina requiere en la entrada las coordenadas X e Y del punto de pantalla y en MD\$ el modo de trabajar (MD\$ = "L" para la lectura del estado encendido/apagado, MD\$ = "S" para la escritura) y proporciona en la salida BT = 1 si el punto está encendido, BT = 0 si está apagado. En la página siguiente se ha representado la ilustración del uso de la matriz B1%(21,3). Como cada elemento de la matriz dispone de 8 bits, ésta puede contener el estado de otros tantos puntos de pantalla; por tanto, los 24 puntos horizontales se memorizan en sólo tres celdas de memoria.

En la página 1665 se ha representado el diagrama de flujo que reconstruye el fondo o presenta el sprite, en función del flag F (F = 1 fondo, F = 2 sprite). Las funciones realizadas son del todo análogas a las de la 2000, con la única diferencia de que se utiliza una matriz diferente en función de F (F = 1 para B1%, F = 2 para S%). En la página 1666 se ha representado el diagrama de flujo de un programa de demostración, en el que las subrutinas 5000 y 6000 (de las que se han omitido los diagramas de flujo) sólo sirven para preparar los datos.

Problemas correspondientes a la animación. En el programa de la página 1666, el sprite está asimilado a un rectángulo que, convertido en imagen binaria, se memoriza en la matriz S%. Esta metodología presenta dos inconvenientes:

- 1 / No permite definir contornos
- 2 / No prevé animaciones alrededor del sprite

El primer defecto tiene como consecuencia la necesidad de llenar toda el área disponible del sprite, que de otro modo aparecería con una zona siempre apagada, correspondiente a la «diferencia» entre la zona del rectángulo efectivamente llenada por el dibujo (sprite) y la total disponible para el sprite.

Este defecto puede superarse con una rutina de presentación más sofisticada. En lugar de limitarse a representar el pequeño rectángulo del sprite, debe construirse la intersección entre sus puntos y los de la pantalla. Es decir, si un punto del sprite está activado, debe presentarse en el monitor (cubre el fondo), mientras que si el punto no está activado, la memoria vídeo debe dejarse sin alteración, como prevista por el fondo. Esto equivale a dibujar el sprite sobre un pequeño rectángulo transparente.

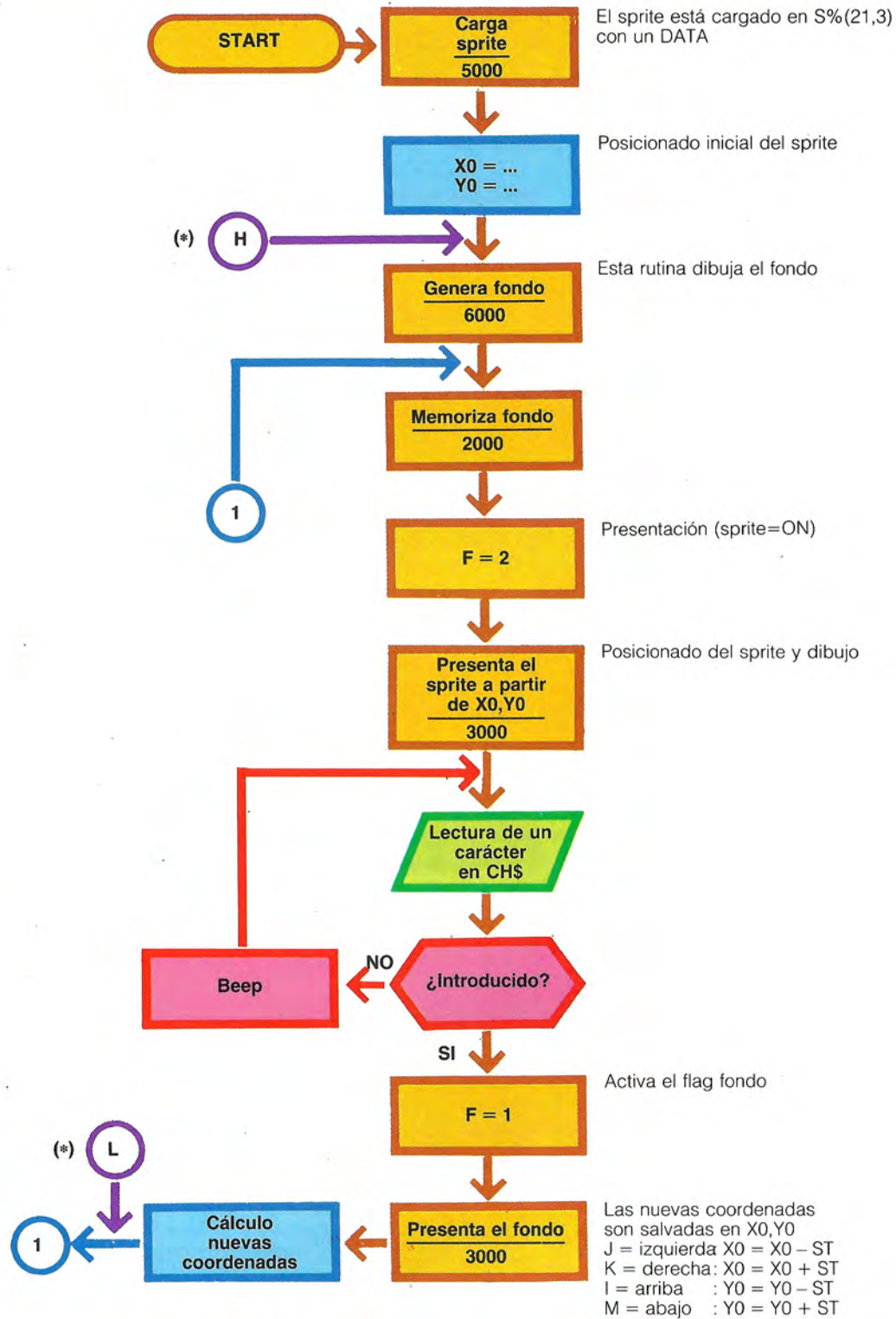
Esta función puede obtenerse muy sencillamente con un IF en fase de transferencia de la imagen sprite a la memoria vídeo; si el punto examinado no está activado, no debe producirse ninguna transferencia, para dejar sin alteración el fondo (ver figura de la página 1671).

En cambio, la solución al segundo defecto es notablemente más compleja, hasta el punto de presentar algunas dificultades incluso en las máquinas que prevén la gestión hardware de los sprites. Efectivamente, se trata de modificar la forma del sprite a medida que el pequeño rectángulo que lo contiene se mueve. El movimiento puede obtenerse de dos maneras:

- 1 / Memorizando algunas posiciones intermedias
- 2 / Esquematisando los desplazamientos con reglas matemáticas

La segunda solución es la más conveniente, tanto para la mayor velocidad de ejecución, como para la menor ocupación de memoria, pero

EJEMPLO DE GESTION DE LOS SPRITES



* A los conectores H y L se hará referencia más adelante.

EJEMPLO DE GESTION DE LOS SPRITES POR SOFTWARE

```

10 REM -----
20 REM      MAIN
30 REM -----
40 DIM S%(21,3),B1%(21,3)
50 ST = 5
60 GOSUB 5000
: REM CARGA SPRITE
70 X0 = 10
: REM Y0 = 180
80 GOSUB 6000
: REM DIBUJA FONDO
90 GOSUB 2000
: REM LEE VENTANA
100 F = 2
110 GOSUB 3000
: REM ESCRIBE SPRITE
120 GET CH$
130 IF CH$ <> "I" AND CH$ <> "J"
    AND CH$ <> "M" AND CH$ <> "K"
    AND CH$ <> CHR$(27) THEN 120
140 F = 1
150 GOSUB 3000
: REM ESCRIBE VENTANA
160 IF CH$ = "I" THEN Y0 = Y0 - ST
170 IF CH$ = "M" THEN Y0 = Y0 + ST
180 IF CH$ = "J" THEN X0 = X0 - ST
190 IF CH$ = "K" THEN X0 = X0 + ST
200 IF CH$ = CHR$(27) THEN END
210 GOTO 90
1000 REM -----
1010 REM CALCULA DIRECCIONES
1020 REM -----
1030 GY = INT (Y / 8)
1040 IF Y < 64 THEN MR = 8192 + GY
    * 128
: GOTO 1070
1050 IF Y < 128 THEN MR = 8232 + (GY
    - 8) * 128
: GOTO 1070
1060 MR = 8272 + (GY - 16) * 128
1070 A = Y - GY * 8
1080 MY = MR + A * 1024
1090 GX = INT (X / 7)
1100 MX = MY + GX
1110 BIT = X - 7 * GX
1120 NN = PEEK (MX)
1130 BY = NN
1140 FOR KK = 0 TO BIT
1150 MS = NN / 2
1160 NN = INT (MS)
1170 NEXT
1180 IF MS <> NN THEN BT = 1
: GOTO 1200
: REM EL BIT ESTA ILUMINADO
1190 BT = 0
: REM EL BIT ESTA APAGADO
1200 RETURN
2000 REM -----
2010 REM LECTURA VENTANA
2020 REM -----
2030 MD$ = "L"
2040 FOR M = 0 TO 20
2050 FOR N = 0 TO 23
2060 X = X0 + N
: Y = Y0 - M
  
```

```

2070 GOSUB 1000
2080 I = 21 - M
      : REM PRIMER INDICE
2090 J = INT (N / 8) + 1
      : REM SEGUNDO INDICE
2100 K = J * 8 - N - 1
      : REM NUMERO DEL BIT
2110 B1%(I,J) = B1%(I,J) + BT * 2 ^ K
2120 NEXT N
2130 NEXT M
2140 RETURN
3000 REM -----
3010 REM ESCRITURA VENTANA
3020 REM -----
3030 MD$ = "S"
3040 FOR M = 0 TO 20
3050 FOR N = 0 TO 23
3060 X = X0 + N
      : Y = Y0 - M
3070 I = 21 - M
      : REM VER 2080
3080 J = INT (N / 8) + 1
      : REM VER 2090
3090 K = J * 8 - N - 1
      : REM VER 2100
3100 IF F = 2 THEN NO = S%(I,J)
      : GOTO 3120
3110 NO = B1%(I,J)
3120 FOR N1 = 0 TO K
3130 N2 = NO / 2
3140 NO = INT (N2)
3150 NEXT N1
3160 IF N2 = NO THEN BB = 0
      : GOTO 3180
3170 BB = 1
3180 GOSUB 1000
3190 IF BT = BB THEN 3220
3200 IF BT > BB THEN POKE MX, BY - 2
      ^ BIT
      : GOTO 3220
3210 POKE MX, BY + 2 ^ BIT
3220 NEXT N
3230 NEXT M
3240 RETURN
5000 REM -----
5010 REM CARGA SPRITE
5020 REM -----
5030 FOR I = 1 TO 21
5040 FOR J = 1 TO 3
5050 READ S%(I,J)
5060 NEXT J,I
5070 DATA 0,0,0
5080 DATA 0,0,0
5090 DATA 0,0,0
5100 DATA 0,0,0
5110 DATA 0,0,0
5120 DATA 0,0,0
5130 DATA 0,0,0
5140 DATA 28,0,248
5150 DATA 62,1,252
5160 DATA 103,3,254
5170 DATA 199,7,158
5180 DATA 143,15,28
5190 DATA 30,15,28
5200 DATA 60,30,60
5210 DATA 60,30,108
5220 DATA 31,252,248
5230 DATA 15,248,144

```

```

5240 DATA 7,240,48
5250 DATA 0,0,0
5260 DATA 0,0,0
5270 DATA 0,0,0
5280 RETURN
6000 REM -----
6010 REM CARGA FONDO
6020 REM -----
6030 HOME
6040 INPUT "NOMBRE DEL FONDO? "; NS$
6050 HGR
      : POKE - 16302,0
      : HCOLOR= 3
6060 PRINT CHR$(4); "BLOAD"; NS$; ",A$20
      "00"
6070 RETURN

```

no siempre puede aplicarse, puesto que difícilmente pueden esquematizarse los desplazamientos de una figura con reglas matemáticas. En cambio, la primera solución sólo requiere un contador y una zona de memoria que conserve todas las posiciones intermedias que se quieren utilizar. Por ejemplo, poniendo en T%(100,21,3) es posible transferir 100 posiciones intermedias (a continuación, en base al contador) a S%(21,3) para tener un movimiento también en el interior del sprite. Una ulterior posibilidad consiste en dibujar sólo la posición inicial y la final, eventualmente con algunas intermedias si las dos son muy diferentes, dejando al ordenador la misión de obtener todos los pasos intermedios necesarios para la animación. Esta técnica todavía está en fase de desarrollo, y en el futuro inmediato se utilizará cada vez más para las producciones cinematográficas.

En la página 1670 se ha representado el diagrama de flujo del programa de la página 1666 modificado para permitir también la animación del interior del sprite. La animación interna del pequeño rectángulo del sprite se obtiene disponiendo un cierto número de formas (10 en el ejemplo) y presentándolas en sucesión. El desplazamiento del sprite todavía va controlado por las teclas, por lo que la animación queda al ritmo de la fase de introducción. Para obtener un resultado más realista puede predefinirse un recorrido y enviarlo automáticamente a ejecución. El método más sencillo para obtener un recorrido (en este caso con 10 posiciones intermedias) consiste en definir una matriz (de 10 valores) en la que se han memorizado (con un DATA) los códigos correspondientes a las teclas de desplazamiento. De esta manera basta con sustituir la instrucción de lectura de teclado con CH\$ = SP(P), donde SP es una matriz que contiene los códigos de desplazamiento.

Un bucle con P de 1 a 10 simulará la presión consecutiva de 10 teclas de desplazamiento y generará automáticamente el recorrido predefinido en el DATA.

La gestión de las colisiones es un parámetro esencial en este género de aplicaciones. Consiste en verificar si un sprite, durante su movimiento, encuentra otro sprite o una zona particular del fondo.

Esta comprobación puede realizarse leyendo la zona de memoria que deberá estar ocupada por el sprite: si la zona es activa se produce una colisión. También en este caso pueden desarrollarse diversas formas de software con diversos grados de sofisticación.

Gestión hardware de los sprites

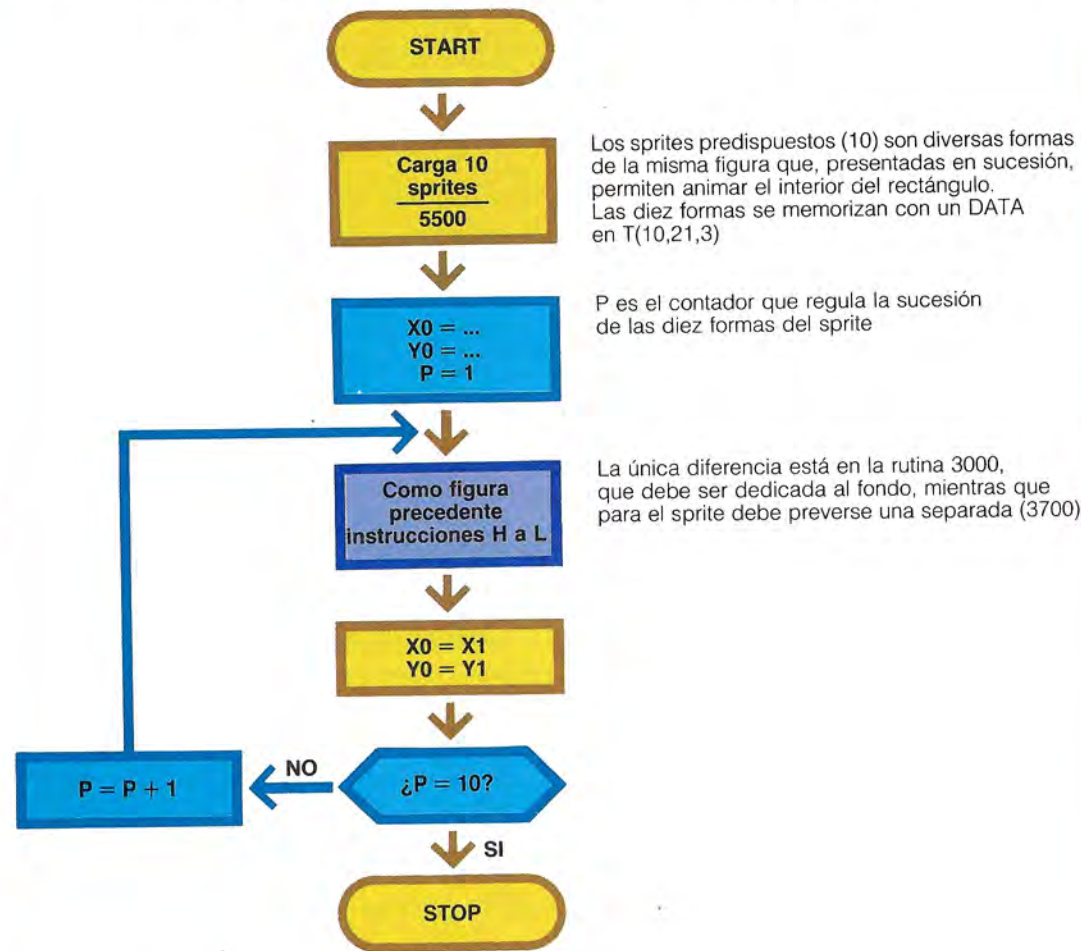
En muchos ordenadores domésticos hay incluidos algunos componentes hardware dedicados a la gestión de los sprites.

Estos componentes pueden memorizar, mover y controlar un cierto número de sprites al mismo tiempo; el usuario sólo debe generar los sprites y el fondo, e impartir las instrucciones del movimiento, aunque sin ocuparse del borrado y de la reconstrucción necesaria de la gestión de los sprites con las instrucciones Basic normales.

En este tipo de máquina, cada sprite está constituido por un cierto número de puntos subdivididos en líneas y columnas; activando algunos puede construirse la forma deseada.

La memorización del sprite. Generalmente, para cada sprite hay previstos algunos centenares de puntos de pantalla. Por ejemplo, en el Commodore 64 hay 504 puntos, divididos en 21 líneas por 24 columnas; por tanto, la ocupación total de memoria es de 63 bytes (cada punto es un bit, por lo que $21 \times 24/8 = 504/8 = 63$ posiciones, ver figura de la página 1671).

GESTION DE LOS SPRITES CON ANIMACION INTERNA



Los sprites predispuestos (10) son diversas formas de la misma figura que, presentadas en sucesión, permiten animar el interior del rectángulo. Las diez formas se memorizan con un DATA en T(10,21,3)

P es el contador que regula la sucesión de las diez formas del sprite

La única diferencia está en la rutina 3000, que debe ser dedicada al fondo, mientras que para el sprite debe preverse una separada (3700)

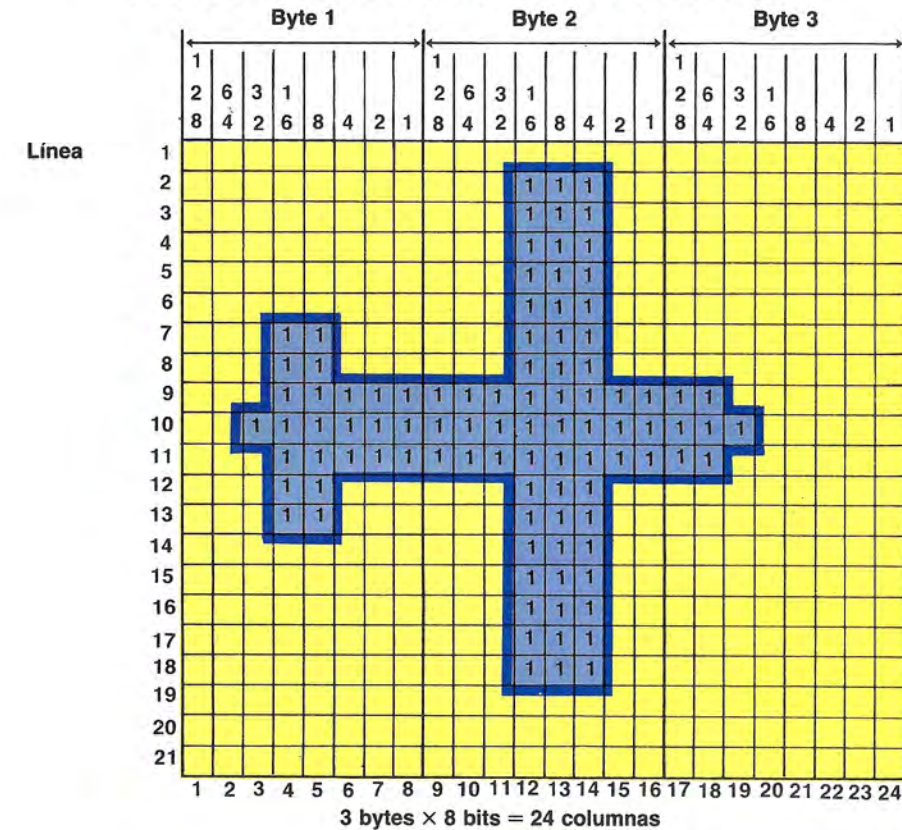
La subrutina 3700 es análoga a la 3000, con las siguientes diferencias:
 — un bucle inicial que transfiere T%(P,21,3) a S%(21,3)
 — eliminado del test sobre F y del bloque de toma del bit K de B1%(I,J); la subrutina ya no es parametrizada y, por tanto, no debe proceder a la gestión del fondo

La construcción de un sprite consiste en memorizar en 63 celdas de memoria contiguas la situación encendido/apagado de cada uno de los 504 bits. Para activar un punto cualquiera debe escribirse en el byte correspondiente un oportuno valor decimal que active el bit seleccionado poniéndolo a 1. Por ejemplo, para activar el punto de cruce entre la primera línea y la primera columna debe escribirse el valor 128, mientras que para la columna 4, el valor decimal es 16 (recuérdese que los valores decimales en función de la posición del bit son 1, 2, 4, 8, 16, 32, 64, 128).

Los valores pueden sumarse, por lo que escribiendo el decimal 7 se activan al mismo tiempo los bits de las posiciones 1, 2 y 3. En la figura de la página siguiente, la primera línea no contiene ningún punto activo y, por tanto, los valores asociados a ésta son 0, 0, 0 (respectivamente bytes 1, 2 y 3).

En la segunda línea, los puntos 12, 13 y 14 son activos y pertenecen al byte 2, valiendo $16 + 8 + 4 = 28$, por lo que la segunda línea está representada por la terna 0, 28, 0. De manera análoga, para el cálculo de los valores en la línea 10 se tiene:

ESQUEMA DE MEMORIZACION DE UN SPRITE



Valores			
Línea	Byte 1	Byte 2	Byte 3
1	0	0	0
2	0	28	0
3	0	28	0
4	0	28	0
5	0	28	0
6	0	28	0
7	24	28	0
8	24	28	0
9	31	255	192
10	63	255	224
11	31	255	192
12	24	28	0
13	24	28	0
14	0	28	0
15	0	28	0
16	0	28	0
17	0	28	0
18	0	28	0
19	0	0	0
20	0	0	0
21	0	0	0

Puntos activos del byte 1: 3, 4, 5, 6, 7, 8 =
 = 32 + 16 + 8 + 4 + 2 + 1 = 63
 Puntos activos del byte 2: todos = 255
 Puntos activos del byte 3: 17, 18, 19 =
 = 128 + 64 + 32 = 224

por tanto, la línea 10 está representada por la terna de valores 63, 255, 224.

La memorización de la tabla así construida puede realizarse con la instrucción normal POKE. Empezando la memorización de la tabla desde la posición 896 (este valor sólo se da a título de ejemplo), la memorización de la primera línea se obtiene con tres instrucciones, una para cada uno de los tres bytes que constituyen la línea:

POKE 896,0 primer byte
 POKE 897,0 segundo byte
 POKE 898,0 tercer byte

mientras que la segunda línea se introduce con

POKE 899,0
 POKE 900,28
 POKE 901,0

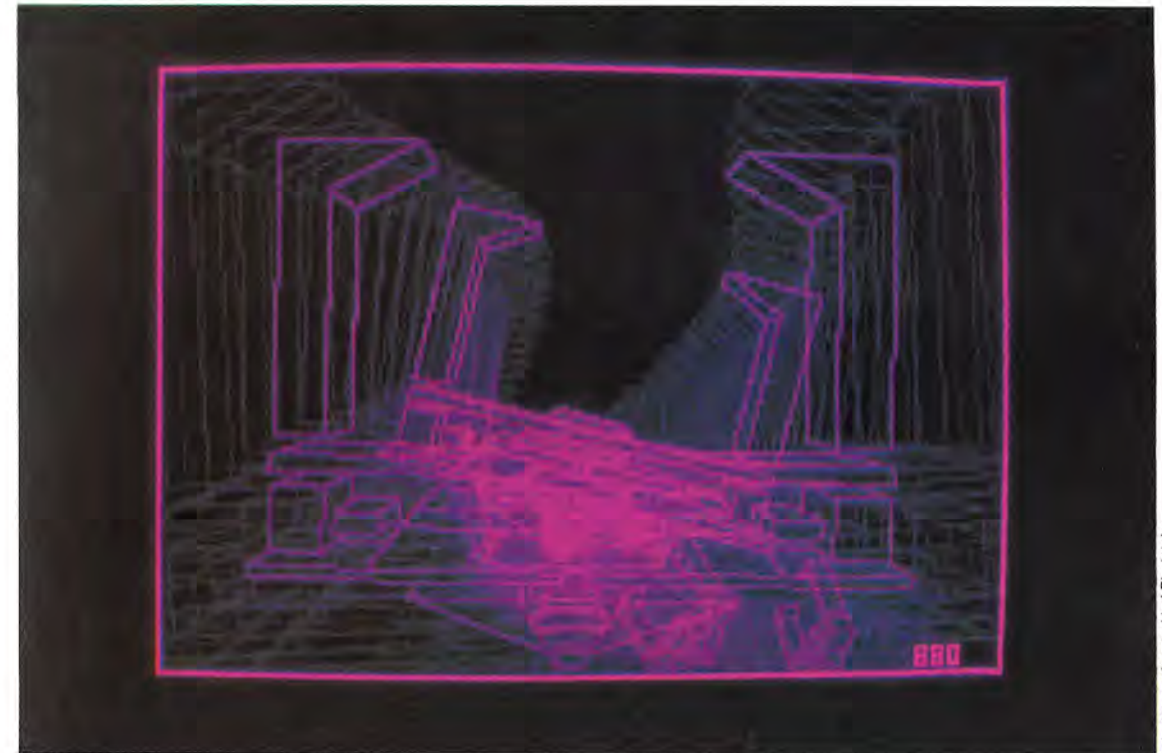
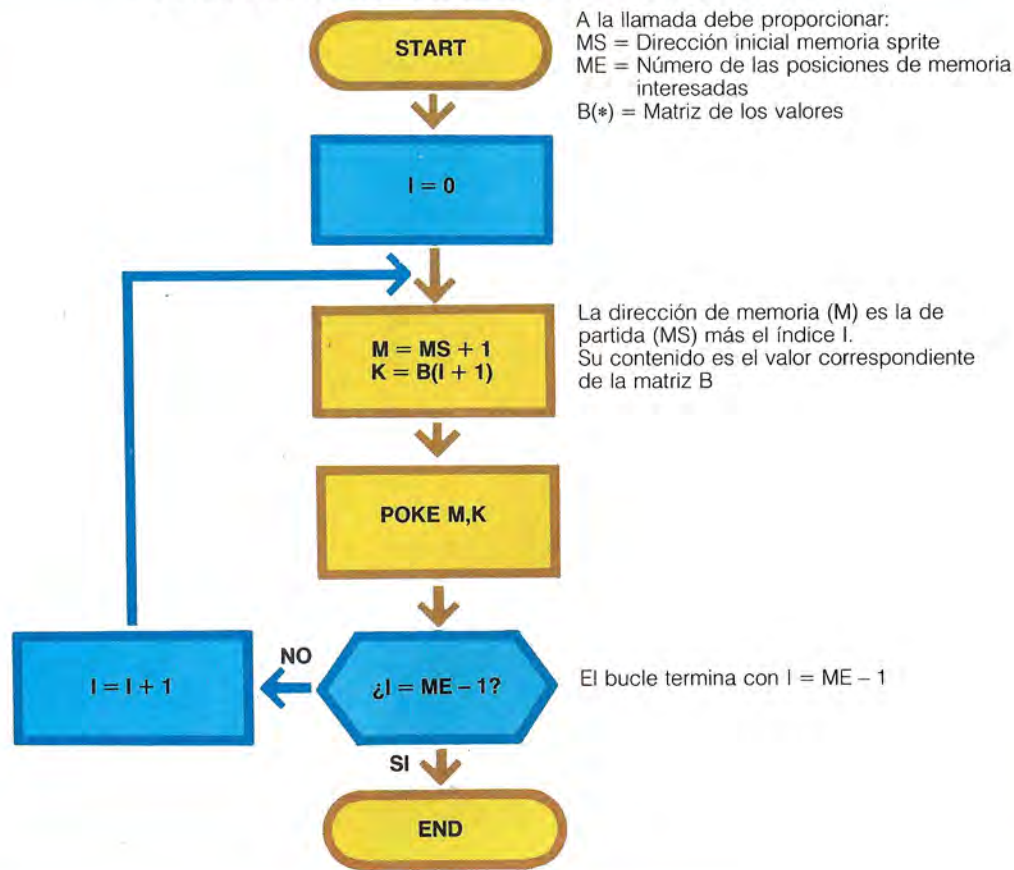
Naturalmente, no conviene escribir tantas instrucciones como cuantas sean las posiciones de memoria a activar. La mejor solución consiste en utilizar un bucle entre la posición de partida y la final.

Abajo se ha representado el diagrama de flujo de una subrutina que memoriza la tabla de la página anterior. Las variables utilizadas son:

B(63) = Contiene los 3 bytes de cada línea (3 bytes × 21 líneas = 63 bytes)
 MS = Memoria de principio
 ME = Número de memorias a activar.
 Para toda la tabla debe implantarse ME = 63 (64 si se tiene en cuenta que a menudo la descripción debe terminar con un byte nulo).

La tabla de definición de un sprite puede me-

EJEMPLO DE MEMORIZACION DE UN SPRITE



P.A. Simon/Graza Net-Photolake

Una imagen animada realizada para el filme "Tron".

morizarse en una zona cualquiera de la memoria RAM, no necesariamente en posición contigua a la definición de otros sprites, puesto que generalmente existe un registro puntero para cada una de ellas. En el Commodore 64, que permite gestionar ocho sprites diferentes (numerados de 0 a 7), los registros punteros a las definiciones de los sprites ocupan las posiciones 2040 a 2047. En las aplicaciones también es útil reservar a la memorización de las tablas que describen los sprites una zona única dividida en tantos bloques de 64 bytes como cuantos son los sprites gestionados. Suponiendo que el área de memoria reservada a los sprites empiece por la posición 832, el sprite definido anteriormente se carga en la posición 896 y en el bloque 2, por lo que después de su activación debe escribirse el valor 14 (12 + 2) en la posición de memoria reservada a la dirección (2040). De esta manera, el sistema puede identificar la posición exacta de memoria a la que se quiere hacer referencia realizando la multiplicación: 14 × 64 = 896.

La presentación del sprite. Al final de la memorización de la tabla que representa el sprite de-

be activarse la lógica para su presentación. Generalmente, la gestión de los sprites se realiza activando o desactivando oportunas y predefinidas posiciones de memoria. Los sprites definidos pueden ser más de uno, y por tanto debe informarse al sistema qué sprite quiere utilizarse. Generalmente, la función se activa poniendo a 1 un bit de una determinada posición de memoria. Por ejemplo, escribiendo 3 (decimal, o sea 11 binario) en dicha posición se declaran activos los sprites cero y uno. En el Commodore 64, el registro que controla la presentación de los sprites está en la dirección 53269. Poniendo a 1 el bit 0 de este registro se activa la presentación del sprite núm. 0, y análogamente se hace para los otros sprites. La presentación sólo puede hacerse si se especifica el punto de la pantalla en que se quiere tener el sprite. Para esta función, cada sprite está controlado por dos registros, en los que es necesario introducir las coordenadas en puntos de pantalla del punto que interesa. Por ejemplo, la posición del sprite 0 está controlada por los registros 53248, 53249: en el primero deberá cargarse (POKE) el valor de la abscisa X, en el segundo el de la ordenada Y.

La presentación del sprite 0 (memorizado en el bloque 2) en el punto 100, 150 se obtendrá del siguiente modo:

```
30 POKE 2040,2
40 POKE 53248,100
50 POKE 53249,150
60 POKE 53269,1
```

El desplazamiento del sprite. El desplazamiento del sprite presentado se obtiene variando con un bucle las coordenadas en los dos registros de control de la posición del sprite. En el caso presentado, el desplazamiento en horizontal del sprite 0 se obtendrá así:

```
100 POKE 53249,100
110 FOR I = 50 TO 150
120 POKE 53248,I
130 NEXT I
```

La línea 100 fija el valor de Y, mientras que el bucle sirve para hacer asumir a la abscisa X los valores comprendidos entre 50 y 150.

Después del desplazamiento de un sprite es necesario asociar un control de las eventuales colisiones con los demás sprites o con el detalle del fondo. También para este control existen particulares posiciones de memoria que indican con su valor si se ha producido una colisión.

En el Commodore 64, la colisión sprite-sprite se controla con el registro 53278, mientras que la colisión sprite-fondo está controlada por el registro 53279. En la figura de la página siguiente se ha representado un ejemplo cualitativo de gestión del desplazamiento de un sprite.

Empleo de la memoria para la gestión de los sprites. En la página 1676 se ha representado el trazado de la memoria del Commodore 64, predispuesto para la gestión de los sprites, que empieza por la dirección 53248.

Las primeras dieciséis posiciones son los registros de posicionado de los ocho sprites que la máquina puede gestionar, en los que deben insertarse las coordenadas del punto de pantalla en que se desea posicionar cada uno de ellos. Sigue un registro (53264) en el que deben memorizarse los bits más significativos de las abscisas X (1 bit por cada sprite). Los cuatro bytes siguientes no se refieren a los sprites, y en la dirección 53269 (correspondiente al 21) se encuentra el registro de activación/desactivación.

Poniendo a 1 un bit de este registro se activa el sprite correspondiente.

En las direcciones 53271 y 53277 se encuentran dos registros que permiten expandir (duplicar) las dimensiones del sprite, respectivamente en la dirección Y y en la dirección X, poniendo a 1 el bit correspondiente (0 para el sprite 0, 1 para el sprite 1, etc.).

El registro 53275 permite fijar la prioridad de un sprite con respecto al fondo. Si por ejemplo el bit 3 de este registro tiene un valor 0, el sprite 3 tiene prioridad sobre el fondo (o sea cubrirá el fondo), si vale 1 sucederá lo contrario.

Las dos posiciones 53278, 53279 contienen los flags de colisión sprite-sprite y sprite-fondo. Si el sprite 2 entra en colisión con el sprite 3, los bits núm. 2 y núm. 3 del registro 53278 quedarán colocados a 1, y análogamente sucederá para el contacto con el fondo.

Para terminar, los registros de 53287 a 53294 son los que permiten especificar el color de los sprites, cargando los códigos numéricos correspondientes (0 = blanco, 1 = negro, etc.).

Los demás registros, bastante numerosos, se destinan a la realización de particulares funciones sobre las cuales no entraremos.

En cambio conviene aclarar con más detalle la gestión de la memorización de los sprites.

Hemos dicho que la descripción de un sprite viene dada por una secuencia de 63 + 1 bytes consecutivos (el último contiene siempre el valor 0) memorizados (POKE) a partir de una dirección de base. Después, la dirección de la posición de base debe memorizarse en el registro que apunta al sprite.

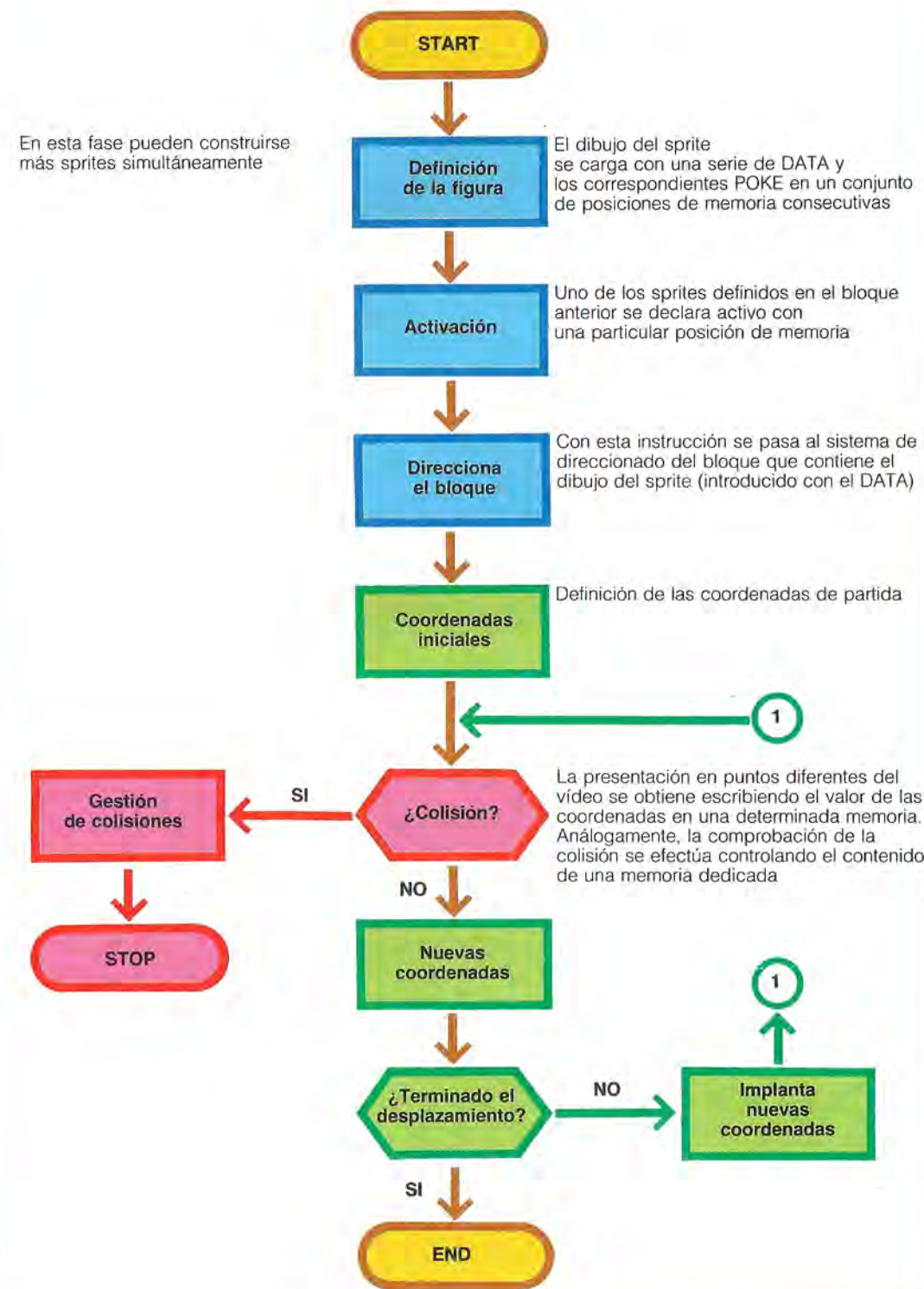
Lo que se memoriza en este último registro no es una dirección de memoria propiamente dicha, sino el número progresivo correspondiente a sucesivos bloques de memoria de 64 bytes cada uno. Multiplicando este valor por 64, el sistema obtiene la dirección efectiva a partir de la cual empieza la descripción del sprite.

La lógica es la de la figura de la página 1677.

Ejemplo de aplicación. En las páginas 1678 y 1680 se ha representado el diagrama de flujo de principio de un juego que utiliza cuatro sprites multicolores.

El juego consiste en desplazar horizontalmente un paracaidista para evitar colisiones con una serie de sprites que aparecen casualmente. El juego termina con la victoria si el paracaidista consigue tocar el suelo.

ESQUEMA LOGICO DE GESTION DE LOS SPRITES



ORDENADO DE REGISTROS SPRITES EN EL COMMODORE 64

Bit. núm.	7	6	5	4	3	2	1	0	
Base = 53248	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
0									SPRITE 0 X
1									SPRITE 0 Y
2									SPRITE 1 X
3									SPRITE 1 Y
4									SPRITE 2 X
5									SPRITE 2 Y
6									SPRITE 3 X
7									SPRITE 3 Y
8									SPRITE 4 X
9									SPRITE 4 Y
10									SPRITE 5 X
11									SPRITE 5 Y
12									SPRITE 6 X
13									SPRITE 6 Y
14									SPRITE 7 X
15									SPRITE 7 Y
53264									Bits altos del valor X
17									
18									TABLA
19									LIGHT PEN X
20									LIGHT PEN Y
53269									Habilitación Sprite
22									
53271									Expansión Y del Sprite
24									Memoria vídeo Sprite
25									Interrupt Request's
26									Interrupt Request MASKS
53275									Prioridad Fondo Sprite
28									Selección Sprite Multicolor
53277									Expansión X del Sprite
53278									COLISION Sprite-Sprite
53279									COLISION Sprite-Fondo
32									INFORMACIONES COLOR
33									Margen
34									Fondo 0
35									Fondo 1
36									Fondo 2
37									Fondo 3
38									Sprites Multicolores
39									SMC0
53287									SMC1
40									Color Sprite 0
41									Color Sprite 1
42									Color Sprite 2
43									Color Sprite 3
44									Color Sprite 4
45									Color Sprite 5
53294									Color Sprite 6
46									Color Sprite 7

EMPLEO DE LA MEMORIA EN LA GESTION DE LOS SPRITES

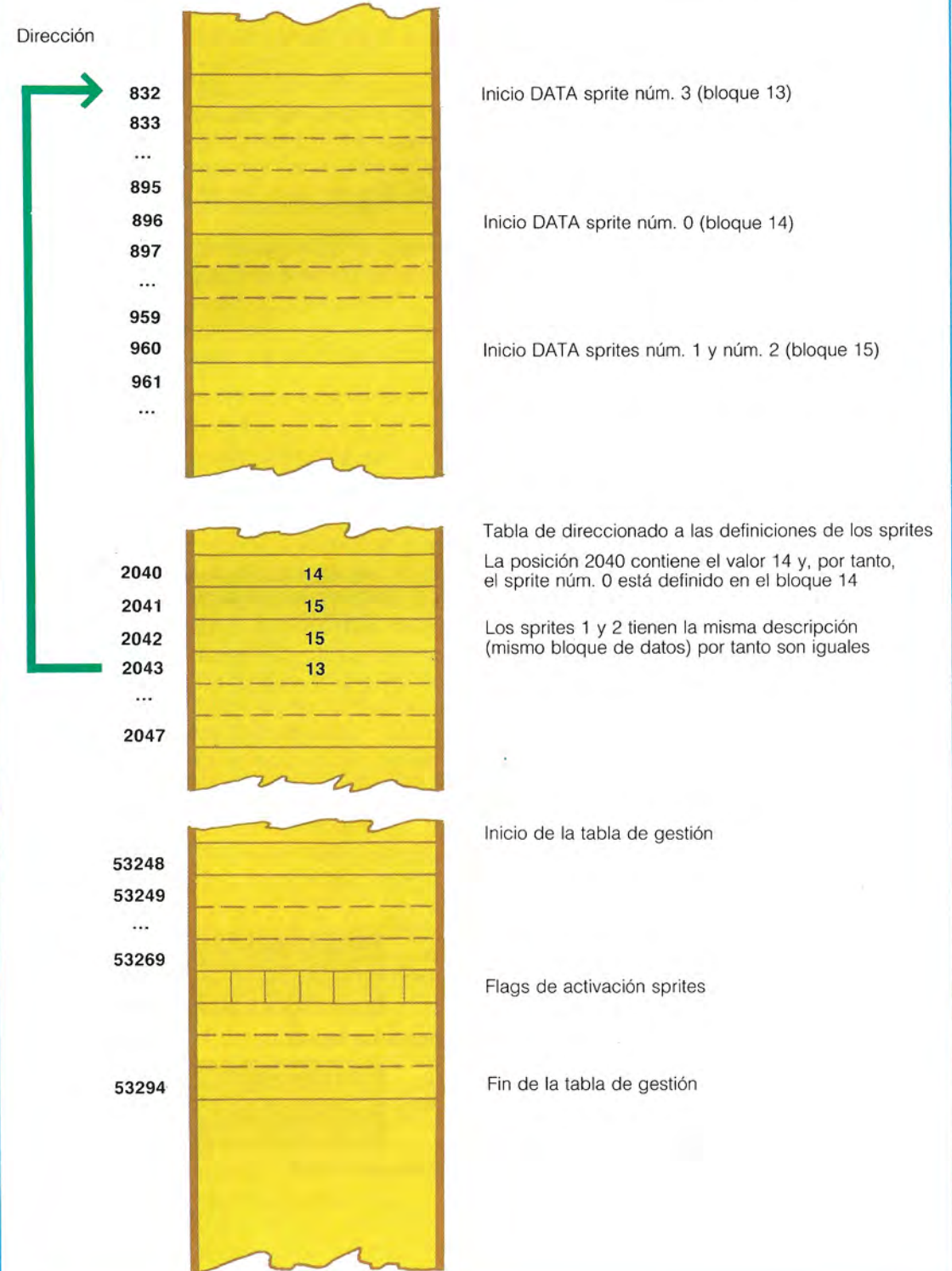
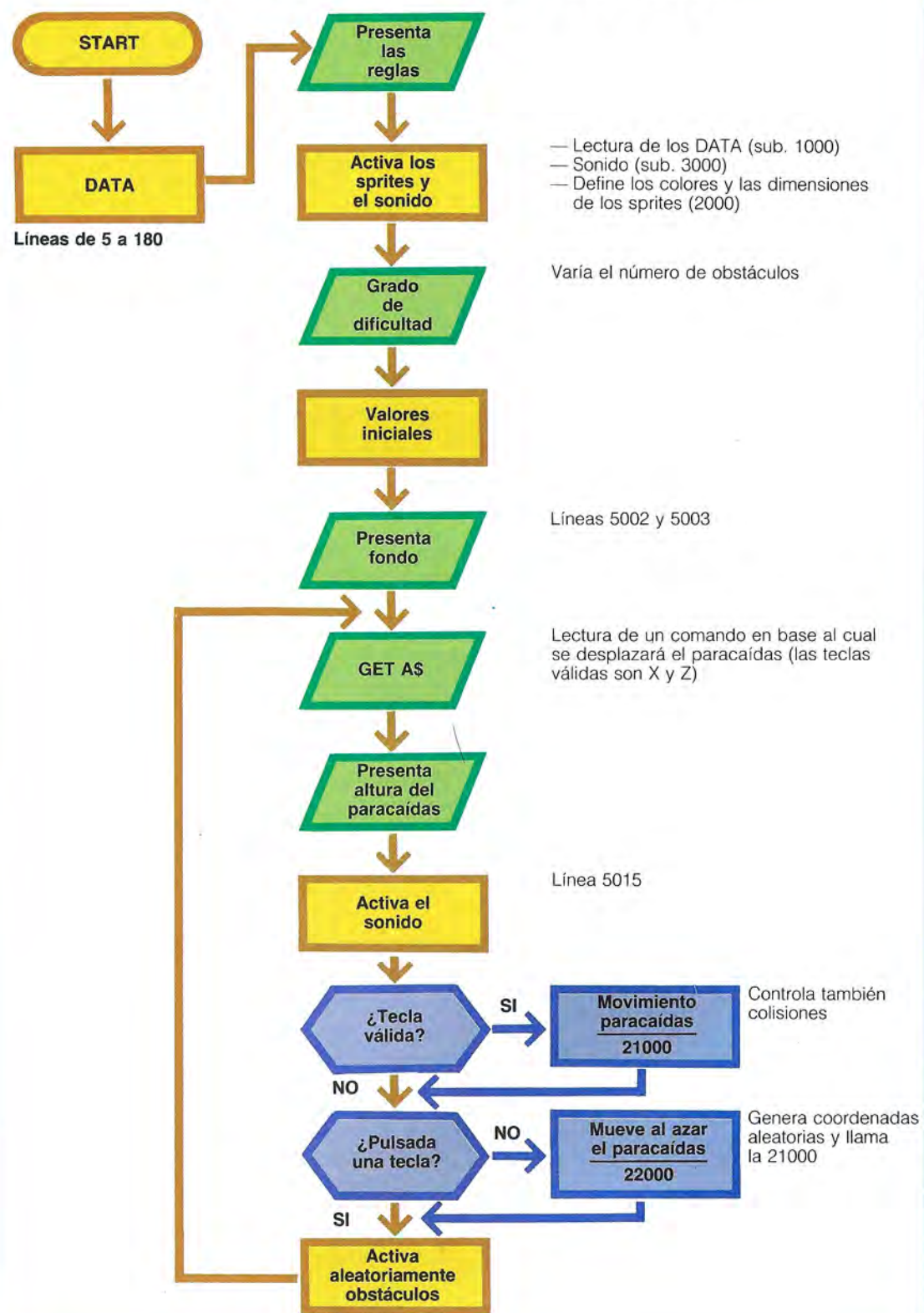


DIAGRAMA DEL PROGRAMA DE PARACAIDAS



El descenso se ve complicado con un viento que desplaza al paracaidista de izquierda a derecha cortando el control al jugador.

Gestión avanzada de los sprites

Además de los videojuegos, los sprites pueden utilizarse también en aplicaciones gráficas. Sin embargo, este uso particular requiere un software de gestión evolucionado que permita llamar todas las funciones necesarias directamente en Basic, sin las complicaciones de los detalles de direccionamiento de la memoria, de ubicación de las tablas, etc.

Las máquinas más modernas, en particular en la categoría doméstica, poseen versiones de Basic avanzado que, además de las instrucciones gráficas, prevén la gestión de los sprites. Las principales funciones implantadas son:

- Definición de los sprites como variable de cadena
- Presentación
- Control de colisiones

Definición de los sprites. Respecto a los casos ya descritos, la metodología es invariable y consiste en reservar una zona de memoria en la que transferir la imagen binaria del sprite. Sin embargo, a diferencia de las formas anteriores, en este caso, cada sprite está definido como una variable de cadena y, por tanto, puede cargarse con una sola instrucción DATA sin necesidad de escribir las tablas de direccionamiento.

Una forma muy usada de esta instrucción es

`SPRITE$(N) = B$`

que define el sprite N igual al contenido de la variable de cadena B\$, a su vez cargada con un DATA.

La instrucción vista, como las siguientes, es propia del sistema Philips VG8000, pero puede utilizarse en todos los ordenadores que emplean el Basic MSX. Esta forma de Basic es muy similar al Basic 80 bajo CP/M, con además numerosas instrucciones dedicadas a los gráficos y al sonido. El Basic MSX es de reciente difusión, pero por la notable facilidad de uso, en particular en las aplicaciones gráficas, se ha adoptado en casi todas las máquinas de reciente construcción basadas en el microprocesador Z80.

Presentación. La instrucción, en la forma más simple, es



Efecto de superposición de imágenes producido con el ordenador.

`PUT SPRITE P, (X,Y),C,N`

donde:

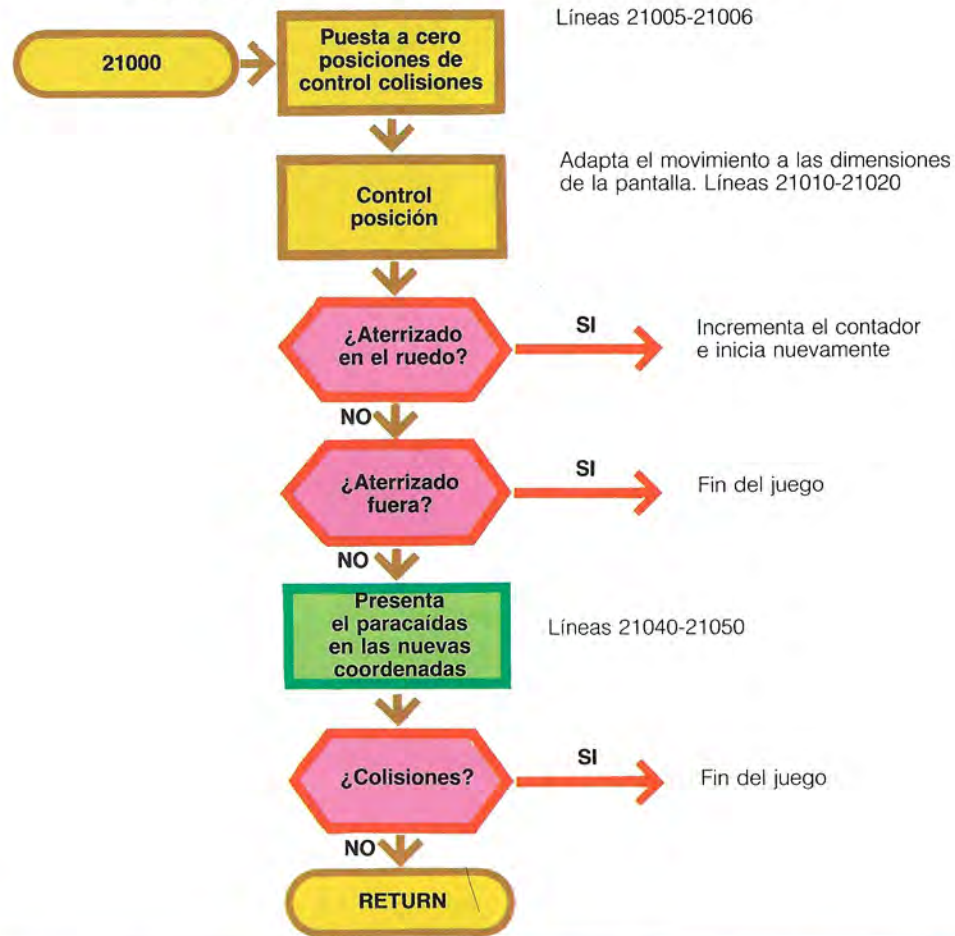
- P indica la prioridad del sprite (para resolver eventuales conflictos que pueden generarse presentando dos sprites en las mismas coordenadas)
- X,Y son las coordenadas
- C es el número que indica el color, comprendido entre 0 y 15
- N es el número de identificación del sprite, definido en la `SPRITE$(N)`

También para esta instrucción, la sintaxis presentada pertenece al sistema Philips VG8000, pero puede utilizarse en todos los ordenadores que contienen el Basic MSX.

Esta instrucción no asigna las dimensiones del sprite, que deben definirse previamente con la instrucción SCREEN, cuya sintaxis es

`SCREEN A, B, C, D`

SUBROUTINA DE DESPLAZAMIENTO Y CONTROL



CARACTERES PARTICULARES UTILIZADOS EN EL PROGRAMA DE PARACAIDAS VERSION CBM 64

CHR \$	Función	CHR \$	Función
147	Borra pantalla	144	Escribe en negro 6
18	Reverse ON	31	Escribe en azul celeste
146	Reverse OFF	5	Escribe en blanco
159	Escribe en azul	17	Desplaza el cursor hacia abajo
28	Escribe en rojo		

Posiciones de memoria

53248 + 29,1	Duplica la dimensión del eje X del segundo sprite núm. 0 (decimal 1)
650,128	Autorepeat
53248 + 28,8	Definición sprite multicolor
53278 } 53279 }	Colisiones
53248 + 42 } 53248 + 37 } 53248 + 38 }	Colores del sprite
	Instrucciones de activación del sprite N. "n": POKE S + 21, PEEK (S + 21) AND "n"

EJEMPLO DE USO DE LOS SPRITES CBM 64

```

5 REM *** MICRO RESQUE CBM 64 ***
10 REM : PARACAIDAS
20 DATA 170,0,2,170,128,2,40,128,14,0,176
30 DATA 47,130,248,43,235,232,170,255,170,170,190,170
40 DATA 186,170,174,168,0,42,161,0,74,1,0,64
50 DATA 113,0,0,243,0,0,40,0,0,40,0
60 DATA 60,0,0,60,0,0,60,0,0,20,0,0,0,0
70 REM :NUBE
80 DATA 60,0,0,255,0,3,255,128,15,191,192
90 DATA 63,255,240,255,255,184,239,254,126,243,253,255
100 DATA 253,243,255,127,247,255,62,111,255,28,63,239
110 DATA 0,31,222,0,15,204,2,7,248,12,15,224
120 DATA 16,31,128,8,127,192,1,239,224,0,126,112,0,56,0
130 REM :HALCON
139 DATA 128,0,1
140 DATA 192,0,3,240,0,15,254,0,127,255,129,255
150 DATA 255,129,255,255,129,255,127,195,254,63,231,252
160 DATA 15,231,240,7,255,224,0,255,0,0,126,0
170 DATA 0,60,0,0,90,0,0,126,0,0,213,0
180 DATA 1,129,128,3,0,192,6,0,96,10,0,80
300 REM ** PRESENTACION **
301 GOSUB 3000
310 PRINTCHR$(147);POKE53281,1:CLR
315 PRINTCHR$(18);CHR$(159);" ";
320 PRINTCHR$(18);CHR$(28);"          MICRO RESQUE  ";
325 PRINTCHR$(18);CHR$(159);" ";
330 PRINTCHR$(17);CHR$(144)"DEBE HACER ATERRIZAR UN GRUPO DE"
331 PRINT "PARACAIDISTAS EN EL RUEDO."
340 PRINT"PERO ATENCION EN NO CHOCAR CON LAS"
350 PRINT"MINAS L;CHR$(120);"1 DURANTE EL DESCENSO"
360 PRINT"Y ATENCION TAMBIEN EN EVITAR LOS"
370 PRINT"MILANOS GIGANTES Y LAS NUBES TOXICAS."
380 PRINT"EL PARACAIDISTA DESCENDERA SOLO": PRINT "POR LA GRAVEDAD"
390 PRINT"PERO DEBE DIRIGIRLO A LA DERECHA Y":PRINT "A LA IZQUIERDA."
400 PRINTCHR$(17);CHR$(18);CHR$(31) "TECLAS DIRECCIONALES: ";CHR$(146);CHR$(144)
410 PRINTTAB(20);"I";CHR$(18);CHR$(28);"X";CHR$(146);CHR$(144);"I...DERECHA"
430 PRINTTAB(20);"I";CHR$(18);CHR$(28);"Z";CHR$(146);CHR$(144);"I...IZQUIERDA"
435 PRINTTAB(12);CHR$(17);CHR$(17);CHR$(18);"F1" PARA INICIO"
440 GETA$:IFA$<>CHR$(133)THEN440
443 PRINTTAB(12);CHR$(18);CHR$(28);" UN MOMENTO... "
445 POKE 650,128:REM AUTOREPEAT A TODAS LAS TECLAS
450 REM :DIRECCION VIC-II
460 S=53248
480 GOSUB 1000
490 GOSUB 3000
500 GOSUB 2000
545 PRINTCHR$(147);CHR$(144):INPUT"DIFICULTAD' ";DF
550 GOTO 5000:REM BLOQUE PRINCIPAL
1000 REM ** LECTURA Y ACTIVACION SPRITES **
1010 FOR D=0 TO 62:REM LEE PARACAIDAS
1020 READ W
1030 POKE 832+D,W:REM SPRITE#3 BLOQUE13
1040 NEXT D
1050 FOR D=0 TO 62:READ W:REM LEE LA NUBE
1060 POKE 896+D,W:REM SPRITE#0 BLOQUE14
1070 NEXT D
1080 FOR D=0 TO 62:READ W:REM LEE EL HALCON
1090 POKE 960+D,W:REM:SPRITES#1E2 BLOQUE15
1100 NEXT D
1120 REM *** ACTIVACION SPRITES ***
1140 POKE S+21,15:REM (DECIMALES=1+2+4+8)
1150 POKE 2040,14:REM NUBE AL BLOQUE 14
1160 POKE 2041,15
1170 POKE 2042,15:REM HALCONES AL BL.15
1180 POKE 2043,13:REM PARAC. AL BL. 13
1200 RETURN
2000 REM DEFINICION SPRITES
2010 POKE S+28,8:REM SPRITE#3 MULTICOL.
  
```

```

2020 POKE S+42,8:REM BIT '10' ANARANJADO
2030 POKE S+37,0:REM BIT '01' NEGRO
2040 POKE S+38,1:REM BIT '11' BLANCO
2050 REM:SPRITE#0 NEGRO Y DOBLE EN X
2060 POKE S+39,0:POKES+29,1
2070 REM:SPRITE#1 COL. CASUAL
2080 POKE S+40,INT(RND(1)*2)+1
2090 REM:SPRITE#2 COL. CASUAL
2100 POKE S+41,INT (RND(1)*7)+7
2140 RETURN
3000 REM ACTIVACION REGISTROS SONIDO
3010 FOR RS=54272 TO 54296: POKE RS,0:NEXTRS
3020 POKE 54274,0:POKE 54275,8:POKE 54278,240:POKE 54277,0
3030 POKE 54296,15
3040 RETURN
4999 REM *** BLOQUE PRINCIPAL ***
5000 SP=0:CS=0:X1=20:X2=20:X3=255
5001 REM
5002 PRINTCHR$(147):PRINTCHR$(142)::GOSUB10000
5003 POKE53278,0:POKE 53279,0
5005 X=INT (RND(1)*150)+100:Y=70:GOSUB21000
5006 V1=0:V2=0:V3=0:NC=0
5010 GET A$
5011 IF Y>100 THEN B$=" ":GOTO 5013
5012 B$=" "
5013 PRINTCHR$(19);CHR$(18);CHR$(158);"ATERRIZADOS: ";PAR;
5014 PRINTTAB(23);CHR$(18);CHR$(5)"ALTITUD: ";(200-Y);CHR$(157);B$;
5015 LB=8:HB=97:HZ=129:GOSUB 30000
5020 A=PEEK(197)
5030 IF A=23THEN X=X+5:Y=Y+3:GOSUB21000
5040 IF A=12 THEN X=X-5:Y=Y+3:GOSUB21000
5045 IF A=64 THEN GOSUB 22000
5050 NC=INT(RND(1)*10)+1
5055 NC1=INT(RND(1)*50)+1
5070 IF NC=20RNC=60RNC=9THENGOSUB 23000
5080 IF NC=30RNC=70RNC=10THENGOSUB 23100
5085 IF NC1=17 AND Y>159 THEN 35000
5090 IFNC=40RNC=8THENGOSUB 23200
5210 GOTO 5010
10000 REM : FONDO
10003 PRINTCHR$(147);:POKE53281,14:POKE53280,6
10020 FORMI=1TO(DF*3)
10025 W=INT (RND(1)*500)+1
10030 POKE 1224+W,88:POKE 55496+W,INT(RND(1)*2)+1
10035 NEXT MI
10036 FORI=1TO21:PRINTCHR$(17);:NEXTI
10037 PRINTCHR$(18);CHR$(30);" ";
10038 PRINTCHR$(18);CHR$(5);" ";CHR$(30);" ";
10040 FORI=1TO2:PRINTCHR$(18):CHR$(30)"
;
10042 NEXTI
10043 PRINTCHR$(19)
10050 RETURN
21000 REM CONTROL COORDENADAS Y PRESENTACION SPRITE#3
21005 POKE 53278,0
21006 POKE 53279,0
21010 IF X<25 THEN X=X+5:RETURN
21020 IF X>240 THEN X=X-5:RETURN
21030 IF X>170 AND X<176 AND Y>197 THEN 2400
21035 IF (Y>197 AND X<170)OR(Y>197ANDX>176)THEN 241000
21040 POKE S+6,X
21050 POKE S+7,Y
21060 REM : COLISIONES
21065 IF (PEEK (53278)AND9)=1THENRETURN
21070 IF (PEEK (53278)AND9)=9THEN24500
21080 IF (PEEK (53278)AND10)=10THEN24520
21090 IF (PEEK (53278)AND12)=12THEN24520
21095 IF (PEEK (53279)AND8)=8THEN26000
21097 IF PEEK (53279)<>8THEN RETURN
21100 REM

```

```

21105 POKE 53278,0
21110 POKE 53279,0
21150 RETURN
22000 REM : CAIDA LIBRE
22010 Y=Y+3
22020 CS=INT(RND(1)*2)+1
22030 IF CS=1 THEN X=X+3
22040 IF CS=2 THEN X=X-3
22050 GOSUB 21000
22100 RETURN
23000 REM MOVIMIENTO NUBE
23005 Y1=INT (RND(1)*50)+100
23007 LB=1:HB=12:HZ=65:GOSUB30000
23010 X1=X1+INT(RND(1)*10)+20
23012 IF X1>=240ANDV1>4THENSP=254:GOSUB25000
23014 IF X1>=240ANDV1<4THENX1=20:GOTO23005
23020 POKE S+0,X1:POKE S+1,Y1:GOSUB21060
23035 RETURN
23100 REM MOVIMIENTO MILANO 1
23102 IF V2<>0 THEN23115
23107 LB=1:HB=12:HZ=65:GOSUB30000
23110 Y2=INT(RND(1)*130)+50:V27+1
23115 X2=X2+INT(RND(1)*10)+10
23117 IF X2>=240ANDV2>4THENSP=253:GOSUB25000
23119 IFX2>=240ANDV2<4THENX2=20:GOTO23110
23120 POKE S+2,X2:POKE S+3,Y2:GOSUB21060
23125 RETURN
23200 REM MOVIMIENTO MILANO 2
23202 IFV3<>0THEN23215
23204 Y3=INT(RND(1)*150)+50:V3=V3+1
23207 LB=1:HB=12:HZ=65:GOSUB30000
23215 X3=X3-10
23217 IFX3<=35ANDV3>4THENSP=251:GOSUB25000
23219 IFX3<=35ANDV3<4THENX3=255:GOTO23204
23220 POKE S+4,X3:POKE S+5,Y3:GOSUB21060
23235 RETURN
24000 REM : HA ATERORIZADO
24030 GOSUB 60000
24050 PRINTCHR$(19);CHR$(18);CHR$(144);" PARACAIDISTA ATERORIZADO
";
24055 PRINTTAB(15);CHR$(18);CHR$(158);"FELICIDADES"
24058 FORT=1TO800:NEXTT
24060 PRINTCHR$(147):PAR=PAR+1:POKE53279,247:GOTO5001
24100 REM FUERA DE BLANCO
24105 PRINTCHR$(19);CHR$(18);CHR$(5);" FUERA DE BLANCO
24110 GOTO 35065
24500 REM : CHOQUE CON OTROS SPRITES
24505 POKE53278,0:SP=247:GOSUB25000:SP=251:GOSUB25000:SP=253:GOSUB25000:SP=254:G
OSUB25000
24510 PRINTCHR$(147);CHR$(18);CHR$(5);"EL PARACAIDISTA HA ENTRADO EN LA"
24515 PRINTCHR$(18);CHR$(5);"NUBE DE ";CHR$(158);"TRIOXINA";CHR$(5);" Y ESTA
MUERTO"
24516 PRINTCHR$(17);CHR$(5);"TOTAL ATERORIZADOS : ";CHR$(18);CHR$(158);PAR
24517 LB=134:HB=24:HZ=65:FORT=1TO3:GOSUB30000:NEXTT
24519 GOTO50000
24520 POKE53278,0:SP=247:GOSUB25000:SP=251:GOSUB25000:SP=253:GOSUB25000:SP=254:G
OSUB25000
24525 PRINTCHR$(147);CHR$(18);CHR$(5);"EL PARACAIDISTA HA SIDO"
24526 PRINTCHR$(18);CHR$(5);"GOLPEADO POR UN:CHR$(158); "MILANO GIGANTE "
24528 LB=1:HB=12:HZ=17:FORT=1TO3:GOSUB30000:NEXTT
24529 PRINTCHR$(17);CHR$(5);"TOTAL ATERORIZADOS : ";CHR$(18);CHR$(158);PAR
24530 GOTO 50000
25000 REM:DESACTIVADOR
25010 POKE S+21,PEEK(S+21)ANDSP
25020 RETURN
26000 REM : CHOQUE CON LAS MINAS
26005 SP=251:GOSUB25000:SP=247:GOSUB25000:SP=253:GOSUB25000:SP=254:GOSUB25000
26010 PRINTCHR$(147);CHR$(18);CHR$(5);"EL PARACAIDSITA HA CHOCADO CON
26012 PRINTCHR$(158);CHR$(18);"MINA ANTIHOMBRE"
26015 LB=1:HB=12:HZ=65:FORT=1TO3:GOSUB30000:NEXTT

```

```

26020 GOTO 50000
26025 PRINTCHR$(17);CHR$(5); "TOTAL ATERRIZADOS : ";CHR$(18);CHR$(158);PAR
30000 REM RUTINA SONIDO
30010 POKE 54276,HZ
30020 POKE 54272,LB
30030 POKE 54273,HB
30040 FOR T=1 TO 5:NEXTT
30050 POKE 54272,0:POKE 54273,0
30060 RETURN
35000 REM VIENTO
35010 SP=254:GOSUB25000:SP=253:GOSUB25000:SP=251:GOSUB25000
35020 FOR K=X TO 255STEP9
35030 PRINTCHR$(19);CHR$(17);CHR$(18);CHR$(5);"VIENTO DE SESENTA NUDOS !!"
35035 LB=134:HB=24:HZ=65:GOSUB 30000
35040 POKE S+6,K:POKE S+7,Y
35045 PRINTCHR$(19);CHR$(17);"
35060 FORT=1TO100:NEXTT:NEXTK
36065 FORT=1TO7:PRINTCHR$(17):NEXTI:PRINTTAB(14);CHR$(18);CHR$(144);"ESTA MUERTO!"
!"
35070 SP=247:GOSUB25000
35080 PRINTCHR$(19);CHR$(5)
35100 END
50000 REM
50010 PRINTCHR$(17);CHR$(5);"VUELVE A JUGAR (SI/NO)";
50020 INPUT R$
50030 IF R$="SI" THEN RUN
50040 PRINTTAB(15);CHR$(17);CHR$(18);CHR$(158);" ADIOS "
50050 FOR T=1 TO 1000:NEXTT:SYS770
60000 REM SONIDO DE ATERRIZADO
60005 POKE 54276,65
60010 POKE 54272,33:POKE 54273,134:POKE 54272,35:POKE 54273,132
60020 POKE 54272,37:POKE 54273,161:POKE 54272,42:POKE 54273,60
60030 POKE 54272,44:POKE 54273,191:POKE 54272,47:POKE 54273,104
60035 POKE 54272,47:POKE 54273,104:POKE 54272,44:POKE 54273,191
60040 POKE 54272,42:POKE 54273,60:POKE 54272,37:POKE 54273,60
60045 POKE 54272,35:POKE 54273,132:POKE 54272,33:POKE 54273,134
60060 POKE 54272,0:POKE 54273,0
60100 RETURN

```

Dos momentos de la ejecución del programa «Paracaídas».



De esta manera, el significado de los parámetros es el siguiente:

- A modo de utilización de la pantalla
- A = 1 texto de 40 caracteres por línea
- A = 2 texto de 32 caracteres por línea
- A = 3 modo gráfico 1 (256 × 192 pixels)
- A = 4 modo gráfico 2 (mismo número de pixels con las figuras constituidas por rectángulos de 4 × 4 pixels)
- B dimensión de los sprites; puede asumir valores entre 0 (sprites de 8 × 8 pixels) y 3 (sprites de 32 × 32 pixels)
- C si está puesto a 1 produce un chasquido cada vez que se pulsa una tecla
- D indica la velocidad de transmisión para el intercambio de datos con el graba-

dor. Los valores admitidos son 1 (1200 baudios) y 2 (2400 baudios)

Control de las colisiones. Las instrucciones que gestionan esta función son dos:

SPRITE ON/OFF/STOP
para activar, desactivar, excluir el control ON SPRITE GOSUB...
llama la subrutina especificada al producirse una colisión.

Abajo y en la página siguiente se ha representado el listado de un programa que muestra el uso de esta instrucción.

Inmediatamente se ve la gran facilidad de escritura de programas con este tipo de Basic.

GESTION DE LOS SPRITES BAJO MSX

```

4 CLS: CLEAR
5 COLOR 5,4,1
7 PLAY"L406CDEFEDCREFGAGFEREDEDGAG"
10 SCREEN 2,2
11 GOSUB 9000
13 C1=15:C2=1:C3=11:P=0
15 REM SPRITE ARAÑA
20 DATA 1,3,7,63,79,159,47,79
25 DATA 147,161,35,5,3,2,4,2
30 DATA 128,192,224,252,242,249,244,242335 DATA 201,133,196,160,192,64,32,64
40 REM SPRITE MARIPOSA ARRIBA Y ABAJO
45 DATA 0,0,1,1,3,7,15,31
50 DATA 63,119,235,221,255,127,61,24
55 DATA 0,0,128,128,192,224,240,248
60 DATA 252,238,215,187,255,254,188,24
65 REM MARIPOSA IZQUIERDA
70 DATA 0,0,0,1,3,7,15,63
75 DATA 63,15,7,3,1,0,0,0
80 DATA 56,124,238,233,191,222,235,254
85 DATA 254,235,222,191,223,238,124,56
90 REM MARIPOSA DERECHA
95 DATA 28,62,119,251,253,123,55,127
100 DATA 127,55,123,253,251,119,62,28
105 DATA 0,0,0,128,192,224,240,252
110 DATA 252,240,224,192,128,0,0,0
140 REM SPRITE HUEVOS
145 DATA 0,0,1,3,3,7,15,31
150 DATA 31,31,31,15,7,0,0,0
155 DATA 0,0,128,192,224,240,248
160 DATA 248,248,248,240,224,0,0,0
170 A$="":B$="":C$="":D$="":E$=" "
200 FOR I=1 TO 32:READ A:A$=A$+CHR$(A):NEXT I
205 SPRITE$(1)=A$
210 FOR I=1 TO 32:READ B:B$=B$+CHR$(B):NEXT I
215 SPRITE$(2)=B$
220 FOR I=1 TO 32:READ C:C$=C$+CHR$(C):NEXT I
225 SPRITE$(3)=C$
230 FOR I=1 TO 32:READ D:D$=D$+CHR$(D):NEXT I
235 SPRITE$(4)=D$
240 FOR I=1 TO 32:READ E:E$=E$+CHR$(E):NEXT I
245 SPRITE$(5)=E$
263 X=100:Y=50:PUT SPRITE 2,(X,Y)C1,2
264 X=1 INT (RND(1)*250)+1: Y1=INT (RND(1)*100)+50:PUT SPRITE 1,(X1,Y1),C2,1
265 J=STICK (1)

```

```

267 IF J=1 THEN Y=Y-7:GOSUB 5000
268 IF J=2 THEN X=X+7:Y=Y-7:GOSUB 5000
269 IF J=5 THEN Y=Y+7:GOSUB 5000
270 IF J=4 THEN X=X+7:Y=Y+7:GOSUB 5010
271 IF J=6 THEN X=X-7:Y=Y+7:GOSUB 5020
272 IF J=3 THEN X=X+7:GOSUB 5010
275 IF J=7 THEN X=X-7:GOSUB 5020
276 IF J=8 THEN X=X-7:Y=Y-7:GOSUB 5020
280 IC=INT(RND(1)*4)+1: IF IC=1 THEN X1=X1-20
282 IF IC=2 THEN X1=X1+20
283 IF IC=3 THEN Y1=Y1-20
284 IF IC=4 THEN Y1=Y1+20
290 GOSUB 1000
295 GOSUB 4000
297 IF X1=X OR Y1=Y THEN 3000
310 ON SPRITE GOSUB 2000
320 SPRITE ON
405 T=T+1
900 GOTO 265
1000 REM MOVIMIENTO ARAÑA
1003 IF X1>250 THEN X1=X1-20:RETURN
1004 IF X1<10 THEN X1=X1+20:RETURN
1005 IF Y1>190 THEN Y1=Y1-20:RETURN
1006 IF Y1<10 THEN Y1=Y1+20:RETURN
1010 PUT SPRITE 1,(X1,Y1),C2,1
1050 RETURN
3000 REM
3005 SPRITE OFF
3010 PLAY "L202BDDC"
3017 COLOR 15,1,1
3018 LOCATE 15,10
3020 PRINT "ESTAS MUERTO"
3025 LOCATE 19,12
3030 PRINT TAB (5);"PUNTOS TOTALES=";P
3050 LOCATE 1,1
3060 INPUT "VUELVES A JUGAR (S/N)";
3070 R$=INKEY$: IF R$="" THEN 3070
3080 IF R$="S" THEN RUN
3200 END
4000 REM APARICION HUEVO
4010 X2=INT(RND(1)*250)+1
4015 Y2=INT(RND(1)*160)+1
4017 IF T>50 THEN 4070
4019 IF T>1 AND T<50 THEN RETURN
4020 PUT SPRITE 5,(X2,Y2),C3,5
4065 RETURN
4070 T=0:RETURN
5000 REM
5001 IF X>253 OR Y>187 OR X<0 OR Y<0 THEN RETURN
5002 PUT SPRITE 3,(X,209),C1,3:PUT SPRITE 4,(X,209),C1,4
5006 PUT SPRITE 2,(X,Y),C1,2: RETURN
5010 REM
5011 IF X>253 OR Y<187 OR X<0 OR Y<0 THEN RETURN
5013 PUT SPRITE 2,(X,209),C1,2:PUT SPRITE 3,(X,209),C1,3
5016 PUT SPRITE 4,(X,Y),C1,4
5018 RETURN
5020 REM
5022 IF X>253 OR Y>187 OR X<0 OR Y<0 THEN RETURN
5023 PUT SPRITE 2,(X,209),C1,2: PUT SPRITE 4,(X,209),C1,4
5026 PUT SPRITE 3,(X,Y),C1,3
5029 RETURN
9000 REM DIBUJO FONDO
9003 LINE (0,0)-(126,98),1:LINE (128,102)-(255,191),1
9005 LINE (128,89)-(255,0),1:LINE (126,102)-(0,191),1
9007 CIRCLE (126,100),2,1
9010 LINE (126,100)-(0,75),1:LINE (128,100)-(255,75),1
9013 LINE (126,101)-(0,125),1:LINE (128,101)-(255,125),1
9015 LINE (127,98)-(86,0),1:LINE (129,98)-(171,0),1
9017 LINE (125,102)-(86,191),1:LINE (129,102)-(171,191),1
9020 FOR I=S TO 140 STEP 6 CIRCLE (126,100),I,1:NEXTI
10000 RETURN

```



GLOSARIO

A

Acceso (Access). Operación de I/O en la unidad de memorización de masa (disco). Se compone de dos fases principales: la búsqueda y la lectura o escritura. En el acceso en el disco flexible, que normalmente no está en rotación, debe realizarse una fase de más consistente en alcanzar la correcta velocidad de rotación. Precisamente ésta es la diferencia que causa la gran diversidad de los tiempos de acceso entre disco y memoria.

Acceso secuencial (Sequential access). Método de lectura/escritura en orden progresivo. En escritura, cada elemento está posicionado físicamente al final del anterior; en lectura, el orden de salida de datos es el de introducción. Referido a un file, el término indica un método de acceso por el que cada operación de I/O procesa el siguiente record.

Acceso selectivo (Random access). Método de acceso a un dato que permite realizar directamente la función. Por ejemplo, un file es de acceso selectivo (random), puesto que un record cualquiera puede ser leído o escrito independientemente de los otros.

También es un método de acceso que permite la lectura o la escritura de cualquier zona sin haber procesado las otras. Puede referirse a un área de memoria (RAM, Random Access Memory) o a un file. A veces se indica con el término acceso directo (direct access).

Acceso serie (Serial access). Método de acceso a un dato que prevé la exploración de todos los que le preceden. Por ejemplo, en un file secuencial, un record puede leerse o escribirse solamente después de los que lo preceden. Generalmente se emplea como sinónimo de acceso secuencial.

Activar (Activate). Operación con que se envía a ejecución un procedimiento o una rutina.

Actualización (Update). Operación de modificación de datos. Generalmente se refiere a un file.

Acumulador (Accumulator). Posición de memoria en la que se depositan los resultados de un cálculo. En los microprocesadores, el acumulador es un registro contenido en el mismo chip que la CPU.

Adaptador (Adaptor). Mecanismo general para adaptar entre sí dos o más componentes. Generalmente es un interfaz que permite la conexión entre dispositivos con conectores diferentes.

Agregado de datos (Data aggregate). Conjunto de elementos bajo un único nombre colectivo. Existen 2 tipos: los vectores y los grupos repetitivos. Un vector, en este contexto, es un conjunto monodimensional de datos ordenados con características idénticas. El segundo tipo indica un conjunto de datos que puede estar presente más veces en el mismo record.

Aleatorio (Random). Se refiere a números generados aleatoriamente con un algoritmo particular contenido en casi todos los lenguajes.

ALGOL. Algebraically Oriented Language. Lenguaje de programación para uso científico, generalmente empleado en los grandes sistemas.

Algoritmo (Algorithm). Está constituido por una serie de reglas o de operaciones matemáticas adecuadas para proporcionar la solución de un problema en un número finito de pasos.

Alimentador (Power supply). Circuito que puede convertir una tensión a los niveles y a la forma requerida por otros circuitos. Por ejemplo, de corriente alterna a corriente continua.

Alinear (To justify). Operación de desplazamiento de los valores en el interior del campo de manera que los espacios queden a una sola parte. Generalmente, los valores numéricos están alineados a la derecha y, por tanto, con los eventuales espacios a la izquierda de los valores, mientras que los datos alfanuméricos están alineados a la izquierda.

Alto nivel (High level). Se refiere a los lenguajes simbólicos orientados al usuario.

ALU. Arithmetic Logic Unit. Unidad lógico-aritmética utilizada en los sistemas de microcalculador como componente dedicado a la realización de cálculos.

Amplitud. Referido a la memoria, indica su capacidad, normalmente expresada en kbytes. Referido a una señal, es el valor que esta señal asume en un determinado momento.

Analogico (Analog). Dispositivo cuyo funcionamiento se basa en una analogía. Los calculadores analógicos, al contrario de los digitales, utilizan todos los posibles niveles de señal, y normalmente, los datos están representados precisamente por estos niveles.

Anidado (Nesting). Operación de englobamiento de una estructura en otra. Por ejemplo, la inserción de un bucle en otro o de una subrutina en otra.

ANSI-COBOL. Lenguaje de programación desarrollado por el American National Standard Institute (ANSI), orientado a los empleos de gestión.

Anticipatory Staging. Técnica para dar velocidad al acceso a las memorias de masa o, en general, a los periféricos lentos. Consiste en desplazar bloques de datos hacia un periférico más veloz incluso antes de que el programa pida su utilización.

APL. A Programming Language. Lenguaje de programación para uso científico, implantado sólo en grandes sistemas.

APT. Automatically Programmed Tools. Lenguaje particular de programación utilizado en la preparación de los datos para el control numérico.

Arbol (Tree). Estructura lógica ramificada que permite, partiendo de un punto cualquiera, llegar al origen (el recorrido es unívoco).

Area. Normalmente se refiere a una zona de memoria. Puede indicar una posición de memoria de masa direccionable, partiendo de una base de datos. En la multiprogramación, cada usuario se asigna un área y la utiliza para los propios procesos.

Argumento (Argument). Valor a pasar a una función, tanto de sistema como definida por el usuario. En algunos casos sólo tiene una función normal y se dice «dummy», o fantasma. Por ejemplo, la instrucción Basic FRE(n) para la lectura del área de memoria disponible utiliza «n» como argumento dummy que, por tanto, puede asumir un valor cualquiera siempre proporcionando resultados válidos.

ASCII. American Standard Code for Information Interchange. Es el código más utilizado para el intercambio de informaciones entre el ordenador y los periféricos. En algunos casos se utiliza también como código de memorización de los datos o de los programas en el disco.

Asignar (Assign). Transferir un valor a una variable. El eventual valor anterior se pierde.

Asíncrono (Asynchronous). Indica un proceso sin una distribución temporal regular. Normalmente se utiliza para indicar una forma particular de transmisión de datos; en general indica un proceso cuya activación no está correlacionada a la ejecución de las instrucciones de programa.

Assembler. El término significa propiamente ensamblador (ver Ensamblador). Por extensión también se llama así al lenguaje ensamblador, muy cercano a la lógica de la máquina, aunque utiliza códigos mnemónicos. De esta manera se pasa a la versión realizable, el lenguaje máquina, utilizando el lenguaje de traducción llamado Ensamblador.

Astable. Circuito que no tiene un estado estable. Puede utilizarse como generador de señales.

Atributo (Attribute). Indica una propiedad característica de uno o más elementos. Por ejemplo, un punto de la pantalla puede tener como atributo el color, un campo de datos de un file puede tener como atributo la condición de ser numérico, el número total de cifras que lo componen y el número de los decimales.

Autohide. Opción existente en los sistemas gráficos más avanzados que permite presentar una figura desde un ángulo particular con borrado de las líneas que no quedan vistas.

Autoplacement. Función del ordenador gráfico, aplicada al proyecto de los circuitos electrónicos, que permite desplazar un componente para optimizar el dibujo del circuito.

B

B-Spline. Representación matemática de una curva plana.

Background. Área de la pantalla que circunda una figura o un carácter (ver también Foreground).

Backspace. Desplazamiento hacia la derecha de un carácter. Es una tecla prevista en muchos teclados; en los que no la poseen, la función se realiza mediante una combinación particular de otras teclas.

Backup. Copia archivada de los datos que se utilizarán en una eventual reconstrucción al producirse un error o una avería que puedan haber causado la pérdida de los datos originales.

Banco de datos (Data bank). Conjunto de datos correspondientes a un tema de la misma naturaleza.

Barker. Forma particular de codificación en la transmisión de datos.

Base de numeración (Base). Número base de un sistema de numeración posicional. Los más usados son: decimal (base 10), binario (base 2), octal (base 8), hexadecimal (base 16).

Batch. Modo particular de realización de un programa que consiste en el proceso diferido con respecto a la introducción de los datos.

Baudio (Baud). Indica el número de estados o de condiciones que se producen en un segundo durante una transmisión de datos. Por ejemplo, al decir 1200 baudios se indica que la transmisión se produce a una velocidad igual a 1200 bits por segundo (un estado o condición es un bit); en los sistemas que utilizan 10 bits para transmitir, un carácter equivale a $1200/10 = 120$ caracteres por segundo.

Baudot. Código de transmisión de datos anterior al código ASCII; utiliza 5 bits para cada carácter.

Benchmark. Conjunto de normas utilizadas para comprobar la calidad de un software o de un hardware.

Biestable (Bistable). Circuito que presenta dos estados estables. Con una oportuna señal puede llevarse al uno o al otro.

Binario (Binary). Condición que sólo tiene dos posibles valores o estados. Normalmente se refiere al sistema de numeración en base 2.

Bisíncrono (Bisynchronous). Método de comunicación que permite la detección de errores múltiples.

Bit. Contracción de binary digit (cifra binaria) y, por extensión, el impulso que la representa y la zona de memoria elemental que puede memorizarla. Puede asumir los valores 0, 1 con el significado OFF, ON.

Bit de paridad (Parity bit). Bit de control cuyo valor indica si los bits anteriores, que constituyen el dato, deben ser en número par o impar.

Blank. Espacio no ocupado por caracteres. En los ordenadores, un espacio en blanco también se considera un carácter, por lo que existe el carácter blank. No debe confundirse con space, que indica un espacio propiamente dicho. Este último término se refiere generalmente a la posición de la cabeza escritora de una impresora o al cursor; no corresponde a ningún dato en la memoria. Y viceversa, el término blank puede indicar una cantidad memorizada.

Blinking. Variación, normal y deseada, de la intensidad de una escritura. Esta forma de presentación se adopta para llamar la atención al operador sobre un mensaje particular.

Bloqueo (Bloking). Combinación de dos o más records que permiten su lectura o escritura con una sola instrucción.

BOM. Bill Of Materials. Término utilizado en las aplicaciones CAD/CAM para indicar la lista de los componentes necesarios para realizar un aparato.

Booleano (Boolean). Un valor lógico que puede interpretarse como «verdadero» o «falso». Generalmente, el estado verdadero se representa por un valor no cero, mientras que el estado falso es cero; sin embargo, en algunos casos puede encontrarse la lógica inversa.

Bootstrap. Indica un programa particular que puede actualizarse. En otros términos, es la versión base de un programa que puede automodificarse. El término también se utiliza para indicar un programa que puede cargarse a sí mismo. Generalmente, el bootstrap es el primer programa cargado en un ordenador, que a su vez permite cargar el sistema operativo. En algunos casos, el bootstrap es residente, por lo que se activa automáticamente a la puesta en marcha de la máquina.

BPI. Bits Per Inch. Densidad de memorización en cinta magnética; indica la cantidad de datos binarios que pueden memorizarse en una pulgada de cinta.

BPS. Bits por segundo. Expresa la velocidad de transmisión (en bits, no en caracteres; para obtener aquélla en caracteres deben dividirse los BPS por el número de bits que forman un carácter).

Bucket. Área de memoria que puede contener más de un record y que puede utilizarse en su conjunto.

Buffer. Ver Memoria tampón.

Bug. Un error en un programa. De este término se derivan: debug, que indica la operación de búsqueda de los errores, y debugger, que indica el programa particular utilizado (el debugger normalmente permite examinar instante por instante los contenidos de memoria y el estado de las variables; este tipo de análisis conduce a la identificación de los errores).

Búsqueda (Search). Proceso de examen para la extracción de un elemento de una lista si satisface determinadas condiciones.

Búsqueda binaria (Binary search). Algoritmo particular para acelerar la velocidad de búsqueda, que consiste en sucesivas divisiones por dos del conjunto a examinar.

Bypass. Indica un elemento de circuito que ofrece un fácil paso a la corriente eléctrica para evitar que pueda circular por otro circuito.

Byte. Grupo de 8 bits, que es la representación en binario de un carácter.

C

CAD. Computer Aided Design. Proceso computerizado para la producción de dibujos industriales. Indica un sistema, entendido como conjunto hardware y software, orientado a la solución de problemas gráficos y de proyecto con el auxilio del ordenador.

CAI. Computer Aided Instruction. Curso de instrucción autodidáctico asistido por ordenador.

Call. Operación con la que se activa un programa o una subrutina; en muchos lenguajes es también un código reconocido.

CAM. Computer Aided Manufacturing. Integración del sistema CAD con sistemas de control y gestión de la producción (por ejemplo, máquinas de control numérico). Sistema de proyecto y de preparación de datos para el funcionamiento de las máquinas de control numérico asistido por ordenador. Es una evolución del CAD.

Campo. Área de un record de datos con un significado específico. Por ejemplo, en el record de un file de datos personales, un campo es el apellido, otro la calle, etc.

Canal (Channel). Conexión a través de la cual pueden fluir los datos.

Capacidad. Componente de circuito que puede acumular una carga. En corriente alterna se utiliza como desacoplo. Se mide en μF (microfaradios).

Capstan. Mecanismo de regulación de la velocidad en los grabadores magnéticos de los grandes sistemas.

Carácter (Character). Cualquier símbolo utilizado como parte de los datos o para su control y organización. No coincide necesariamente con un carácter del alfabeto, puesto que para el ordenador se consideran caracteres también los códigos particulares utilizados para la gestión de los periféricos o como controles en los intercambios con otros sistemas (ver tabla códigos ASCII).

Carácter alfabético (Alphabetic character). Indica una letra del alfabeto. En los ordenadores, normalmente se reconoce el alfabeto inglés.

Carácter alfanumérico (Alphanumeric character). Indica un conjunto de símbolos constituidos tanto por letras del

alfabeto como por cifras, entendidas como símbolo y no como valor.

Carácter de control (Control character). Un carácter cualquiera (en sentido llano, no sólo alfabético) utilizado para realizar particulares funciones como la activación, la modificación o la finalización de una operación de control (transmisión de datos, memorización en disco, etc.).

Catalog. Lista de los files (datos, programas) presentes en disco.

CCITT. Comité Consultivo Internacional para Telefonía y Telegrafía.

Celda. Conjunto de posiciones elementales de memoria (bits) contiguas, generalmente con contenidos homogéneos. En las Bases de Datos, la definición de un conjunto de celdas es la de no cruzar subdivisiones mecánicas sobre la memoria de masa, como por ejemplo los sectores.

Cellular splitting. Técnica particular utilizada para disponer los records adjudicados a un file. Consiste en organizar los records en una celda dividiéndola en dos partes cada vez que se llena.

Ciclo (Cycle). Indica una fase de proceso de la CPU, por ejemplo, la toma de un dato. En general, una instrucción está compuesta de varios ciclos.

Cilindro (Cylinder). Define un área en el disco que puede ser leída sin movimiento del mecanismo de acceso. Generalmente, un cilindro es el conjunto de dos pistas, una sobre cada cara del disco, que pueden ser leídas sin desplazar el par de cabezas.

Cinta perforada (Punched tape). Ficha de cartulina sobre la que se escriben los datos en forma de perforaciones. La presencia de un orificio en una determinada posición indica el estado del bit correspondiente.

Circuito impreso (Printed board). Técnica de realización de las conexiones eléctricas utilizando métodos de fotoinscripción sobre una placa de material aislante recubierta de cobre.

Circular file. Un tipo particular de file en el que los nuevos records que deben insertarse reemplazan a los anteriores.

CL file. Cutter Location file. Conjunto de datos en la salida de un sistema gráfico que constituyen las coordenadas para el control de una máquina de control numérico.

Clave (Key). Campo de datos utilizado para la búsqueda y el acceso al record. La clave puede ser primaria, cuando identifica unívocamente un record, o secundaria, que en este caso puede estar compartida por más records. Parte del record que permite la identificación. En los ordenamientos, la clave es el campo en base a cuyo contenido se realiza la comparación entre los diversos records para obtener una salida ordenada.

Clipping. En electrónica, indica la eliminación de la parte más alta de la señal. En las aplicaciones gráficas en ordenador, a veces se ha utilizado para indicar el procedimiento por medio del cual se elimina una parte del dibujo para presentar en la pantalla objetos que, de otro modo, no serían visibles.

CodasyI. Conference of Data Description Languages. Es la organización encargada de actualizar o de emitir nuevas especificaciones de Cobol.

Codificador (Encoder). Circuito utilizado para codificar datos.

Código (Code). Una representación de datos o de programas que puede estar comprendida en el ordenador (por ejemplo, el código ASCII). De este término deriva el uso para indicar la operación de escritura de un programa con el término «codificación», es decir la traducción de las funciones a realizar en forma comprensible para el ordenador. Ver también Flag.

Código Huffman (Huffman code). Forma particular de codificación en la que los caracteres utilizados con mayor frecuencia están representados con un número de bits inferior al usual. Esta técnica, notablemente compleja, es útil para disminuir el espacio ocupado en la memoria de masa.

Cola (Queue). Lista de elementos en espera de ser procesado. Por ejemplo, la lista de las peticiones de servicio por parte de periféricos o de procedimientos a enviar a ejecución.

Coma fija (Fixed point). Representación de valores numéricos reales con la coma en posición definida.

Coma flotante (Floating point). Representación de los valores numéricos reales en forma de mantisa y exponente.

Comando (Command). Orden emitida para utilizar una determinada función del sistema operativo o del lenguaje empleado.

Comentario (Remark). Una línea de programa que tiene el único objeto de proporcionar explicaciones. La posibilidad de incluir líneas de comentario existe en todos los lenguajes simbólicos. Generalmente, el Compilador procede a eliminarlas de manera que no ocupen inútilmente memoria en la versión realizable (la versión realizable se expresa en códigos numéricos y, por tanto, los comentarios no tendrían ningún sentido).

Compilador (Compiler). Programa que puede traducir otros programas, escritos en lenguaje simbólico, en una forma comprensible para el procesador. El Compilador, a diferencia del Intérprete, realiza la traducción de todo el programa y genera una forma intermedia, llamada reubicable, que todavía no puede enviarse a ejecución.

Complemento (Complement). En general indica un opuesto. En particular es un número que puede obtenerse como resta entre otros dos números dados. Por ejemplo, el

complemento a 10 de 7 es 3 ($10 - 7 = 3$). En los ordenadores se utiliza frecuentemente el complemento a 2, base de numeración.

Complemento a dos (Two's complement). Forma de representación de los números negativos en el sistema binario.

Compresión (Compression). Método que permite la reducción de la longitud del campo clave utilizando adecuados algoritmos de transformación. Operación apta para reducir el espacio ocupado por los datos, por ejemplo, eliminando los espacios vacíos.

Comunicación (Communication). Operación de transmisión y recepción de datos. La comunicación puede realizarse de manera directa si es a distancias cortas, por ejemplo entre el ordenador y una impresora u otro periférico local; y viceversa, para grandes distancias se necesita el uso de aparatos auxiliares, como el modem.

Concatenado (Concatenation). Unión de dos cadenas para formar una tercera, suma de las anteriores. También puede indicar la unión de dos programas o agrupaciones de datos en la memoria de masa.

Constante (Constant). Cualquier valor físico, numérico o literal.

Constructor. Término utilizado en Pascal para indicar una lista de valores empleados como asignación a un conjunto.

Contador. Circuito que puede contar los impulsos recibidos.

Control numérico (aparato de). Aparato de producción automatizada para realizar un determinado trabajo bajo el control de datos numéricos, generalmente obtenidos por sistemas CAM.

Control numérico (flujo de). Instrucciones memorizadas, generadas por ejemplo con sistemas CAM, para controlar máquinas con el ordenador.

Conversacional (Conversational). Método de introducción de datos consistente en proponer, por parte del ordenador, una serie de éstos esperando y validando las respuestas.

Coordenadas (Coordinates). Serie de números que permiten identificar un punto con respecto a un sistema de referencia. Pueden referirse al plano y, en tal caso, son dos (x,y), o al espacio (tres valores). Por ejemplo, se utilizan para identificar la posición del cursor en la pantalla o la de la pluma del plotter.

Coordenadas absolutas (Absolute coordinate form). Especifican la posición de un punto con respecto al origen del sistema de referencia utilizado.

Coordenadas relativas (Relative coordinate). Sistema de identificación de un punto por medio de los desplazamientos de otro punto cualquiera.

Correlador (Correlator). Circuito cuya salida vale 1 si todas las entradas son 0. Se utiliza para formación de códigos.

CPU. Central Processing Unit. Componente hardware que controla el funcionamiento de las demás unidades, realiza cálculos y gestiona los periféricos.

CR. Sigla que indica el código de retorno de carro (Carriage Return). Enviando este código (ASCII) a una impresora o a un vídeo, se tiene el posicionado de la cabeza de escritura (o del cursor) en la primera posición útil de la misma línea; para cambiar de línea, pasando a la siguiente, también debe enviarse el código LF. Por tanto, en general, al final de una escritura siempre habrá presentes los códigos CR y LF.

Cristal (Crystal). Generalmente se refiere al cuarzo (cristal de cuarzo). Está constituido por una fina lámina de cuarzo adecuadamente mecanizado, y tiene la propiedad de entrar en oscilación si se excita eléctricamente. Aprovechando estas oscilaciones, muy estables y precisas, pueden construirse osciladores (generadores de señales o de reloj) de alta calidad.

Current Loop. Método hardware de transmisión de datos. La señal está constituida por una corriente típicamente comprendida entre 20 y 50 mA.

Cursor. Señal de identificación de una posición en la pantalla. Generalmente está representado por un pequeño trazo o por un rectángulo luminoso.

CH

Checkpoint. Un método particular que consiste en escribir en un file los parámetros de un programa mientras se realiza. Si el programa se interrumpe por causas accidentales, puede reemprenderse desde el punto definido en la última memorización (checkpoint). Esta técnica se utiliza en procedimientos que tienen un considerable tiempo de proceso para evitar tener que volver a empezar desde el principio ante una interrupción.

Chip. Plaquita de semiconductor sobre la que se ha realizado un circuito integrado. Por extensión, sinónimo de circuito integrado.

D

DASD. Direct Access Storage Device. Sigla utilizada para identificar un dispositivo de memoria de acceso directo.

Data Base. Conjunto de datos correlacionados entre sí que pueden servir para diversas aplicaciones. En los sistemas más evolucionados, la forma de memorización de los datos se hace de manera que sean independientes del programa que los utiliza. El conjunto de los programas necesarios para su gestión toma el nombre de Data Base Management System. Generalmente, estos programas utilizan lenguajes

dedicados a este fin, como el Data Description Language y el Data Manipulation Language.

Data-set. Conjunto de valores que constituyen un bloque de datos.

Datos (Data). Informaciones a proporcionar antes del proceso. Los datos deben ser todos conocidos.

DCE. Data Communication Equipment. Sigla que indica los aparatos generales que sirven para realizar una conexión de transferencia de datos. Los aparatos conectados con el DCE toman el nombre de DTE (ver).

Debug. Operación de búsqueda y eliminación de los errores (ver Bug).

Declaración (Declaration). Conjunto de instrucciones que definen el tipo o las dimensiones de las variables o los valores de las constantes. Puede ser explícita, si se indica con líneas de programa, o implícita si es asumida automáticamente por el lenguaje particular. Por ejemplo, la declaración de un dimensionado en Fortran es explícita, mientras que en Basic, dentro de ciertos límites, puede ser implícita.

Decodificador (Decoder). Circuito utilizado para decodificar datos.

Decodificar (To decode). Convertir un valor de la forma codificada, o sea en código, a la forma normal.

Delimitador. Carácter de separación entre valores. Por ejemplo, la coma en algunos lenguajes es un delimitador durante la introducción de datos.

Demand staging. Proceso que consiste en mover un bloque de datos desde un periférico lento a otro rápido, cuando el programa requiere su uso. Es una técnica opuesta al método Anticipatory Staging.

De Morgan. Teorema utilizado en el análisis de los circuitos digitales para convertir esquemas con lógica positiva en los equivalentes con lógica negativa y viceversa.

Dependencia funcional (Functional dependence). Se tiene una dependencia funcional cuando un cierto atributo de una relación depende de otro atributo de la misma relación y la dependencia es unívoca. En una situación similar se dice que el primer atributo identifica al otro.

Desbordamiento (Overflow). Referido a un programa, indica una superación general de la capacidad de memoria. Para los files indica una área particular donde se memorizan los records que no pueden colocarse en la zona lógica a la que pertenecen.

Deshabilitado (Disabled). Estado que no permite la acogida de algunas funciones. Por ejemplo, la deshabilitación de las interrupciones no permite interrumpir el programa.

Desviador (Switch). Indicador que en virtud de su valor provoca una variación en el flujo del proceso.

Diagnóstico (Diagnostic). Operación con la cual se informa al operador acerca de las posibles causas de un error. Existen dos tipos fundamentales: el diagnóstico de sistema y el de los programas de aplicación. El primer tipo está gestionado por el sistema o por los Compiladores, y evidencia generalmente errores formales o de sintaxis. El segundo debe ser preparado por el programador en función de la aplicación particular.

Diagrama de flujo (Flow chart). Representación gráfica de las funciones a realizar en un programa.

Diccionario (Dictionary). Catalogación de todos los tipos de datos que proporcionan su nombre y la estructura.

Digital. Indica un dispositivo que sólo puede reconocer estados definidos y no valores intermedios. Generalmente, los dos posibles estados están identificados con 0 y 1 (algunos dispositivos electrónicos son de tres estados; el tercero no es un estado lógico, sino sólo una condición eléctrica particular en que está conectado el dispositivo).

Digitalizador (Digitizer). Aparato que convierte datos en forma numérica comprensible para el ordenador (digital).

Dinámico. Acontecimiento que se produce durante la ejecución de un programa. La ubicación de una cadena es un acontecimiento de tipo dinámico.

DIP. Dual Inlined Package. Cápsula con patillas dispuestas en dos filas paralelas que encierra el chip.

Dirección (Address). Especifica un registro, una posición de memoria o un dispositivo en general. Generalmente está compuesto de 8 bits; en las máquinas mayores puede ser de 16 o de 32 bits. Su longitud indica la extensión de memoria utilizable por el sistema.

Direccionamiento. Envío a la memoria.

Directorio (Directory). Tabla de identificación o de correspondencia de un conjunto de datos. El directorio de un disquete es el área, reservada al sistema, en que se escribe el nombre de cada file y su posición (además de otras informaciones que dependen del sistema).

Diskpack. Conjunto de discos magnéticos, utilizado solamente en los grandes sistemas.

Display. Componente que puede presentar visualmente informaciones.

Disponibilidad (Availability). Es la medida de la capacidad de un sistema para ser utilizado.

Divisor (Divider). Circuito a cuya salida hay un impulso para cada «n» impulsos de entrada. Normalmente está realizado con una configuración particular de contadores.

DL/I. Lenguaje particular, implantado en máquinas IBM,

para la definición lógica y física de las estructuras de datos.

DMA. Direct Memory Access. Transferencia de datos entre un periférico y la memoria obtenida sin interesar la CPU.

Doble precisión (Double precision). Método de memorización de datos que se sirve de un espacio de memoria doble del normal para utilizar más cifras significativas.

Dominio (Domain). Conjunto de datos del mismo tipo ligados por una relación.

DOS. Disk Operating System (Sistema operativo disco). Es el conjunto de programas y subrutinas que constituyen un sistema operativo memorizados en disco. El conjunto de esos programas puede dividirse en dos grupos fundamentales: los transeúntes y los residentes. Los primeros se cargan en memoria sólo cuando se necesita su activación y los otros siempre están presentes; por ejemplo, los módulos de gestión de los periféricos que deben realizarse en todos los programas deben ser residentes.

Dot. Indica un punto, generalmente de la pantalla; puede considerarse como sinónimo de pixel (ver).

DRC. Design Rules Checking. Procedimiento de verificación de datos para evidenciar algunos tipos de error preestablecidos, por ejemplo incompatibilidad de las tolerancias.

DTE. Data Terminal Equipment. Sigla que indica los aparatos generales conectados entre sí en una red.

DTL. Diode Transistor Logic. Circuitos lógicos obtenidos combinando diodos y transistores.

Dummy. Valor utilizado sólo para realizar determinadas reglas formales pero que no se considera. Por ejemplo, un argumento en una función puede ser dummy en cuanto no se emplea realmente para realizar cálculos, sino sólo como regla formal (en realidad, el argumento dummy lo utiliza el sistema, aunque no como valor).

Dump. Operación de emisión, en impresora o en vídeo, del contenido de una zona de memoria o de una parte del disco sin ningún proceso.

Duplex. Modo de transmisión en los dos sentidos, simultáneo. También se indica con el término full duplex para distinguir de la modalidad half duplex que no prevé la simultaneidad.

E

EBCDIC. Extended Binary Coded Decimal Interchange Code. Código particular utilizado en la transmisión de datos como alternativa al código ASCII más conocido. En el código EBCDIC, cada carácter está representado por 8 bits, el primer valor es 0000 0000, que representa el espacio en blanco (SP), y el último es 1111 1001, al que corresponde el carácter 9.

ECL. Emitter Coupled Logic. Familia particular de circuitos integrados de uso poco frecuente, puesto que son difícilmente acoplables con los de otras familias.

Eco (Echo). Retransmisión del dato recibido. Por ejemplo, los caracteres introducidos por teclado se presentan como eco en el vídeo.

Edición (Edit). Modificación de datos. En particular, el término se utiliza para indicar las operaciones de corrección (inserción o modificación) de los programas.

Editar (To edit). Puede indicar la operación de preparación de los datos para la impresión o la operación de modificación de un programa; en este caso, generalmente se emplea el término «edición».

EDP. Electronic Data Processing. Edición electrónica de datos.

EIA. Electronic Industries Association.

Ejecución (Execute). Realización de una instrucción.

Electrolítico. Referido a los condensadores, indica un tipo particular con el que pueden realizarse elevados valores de capacidad. Los condensadores electrolíticos se utilizan principalmente en los alimentadores.

Elemento (Element). Un dato particular en una lista o en una matriz. Un término de un grupo, por ejemplo un valor particular en una matriz.

Elementos finitos (Finite element). Método de cálculo utilizado principalmente en ingeniería mecánica.

Embedded pointer. Identifica un tipo particular de gestión que prevé los punteros a los datos contenidos en el mismo record de datos así como en el directorio.

Ensamblador (Assembler). Programa de sistema que traduce en código máquina las instrucciones en lenguaje Assembler.

Entero. Número sin decimales.

EOF. End Of File. Señal de reconocimiento del final de un file.

ES. Electrical Schematic. Dibujo de un aparato obtenido utilizando los símbolos convencionales para representar los componentes.

Escala (Scale). Relación entre las dimensiones de una imagen y las del objeto real.

Escalar (Scalar). Magnitud que no posee una orientación, o que no es una matriz.

Escape. Código utilizado en muchos sistemas operativos para indicar una acción particular a realizar. Normalmente, corresponde a una tecla.

Esquema de bloques (Flow chart). Ver Diagrama de Flujo.

Esquema canónico (Canonical schema). Modelización de un conjunto de datos que representa la estructura de los propios datos, independientemente tanto del uso que se deberá hacer de ellos como del hardware o del software particulares que lo emplearán.

Etiqueta (Label). Indicador, numérico o literal, que sirve para direccionar un proceso o una instrucción; en general constituye una dirección.

Eurístico (Heuristic). Método de solución de un problema que consiste en realizar diversos intentos analizando los errores.

Exactitud (Accuracy). En general indica la capacidad de detectar diferencias.

Execute. Indica el ciclo de ejecución de una instrucción que normalmente sigue al fetch.

Exploración (Scan). Operación con la que se examina secuencialmente una determinada zona. Por ejemplo, el movimiento del haz electrónico sobre la pantalla vídeo (ver Raster scan).

Extent. Espacio en el disco reservado a un file.

F

Factor de bloqueo (Blocking factor). Número máximo de records previstos en un bloque; generalmente se utiliza en Cobol.

Factor de escala (Scale factor). Valor multiplicativo de las dimensiones de un dibujo para introducirlo dentro de las medidas del vídeo o del plotter.

Fan-In. Término con el que se expresa la carga generada por un circuito en su entrada. Proporciona un orden de magnitud de la potencia necesaria para controlar el circuito.

Fan-Out. Indica la capacidad de suministro de un circuito, es decir, el orden de magnitud de la potencia que puede suministrar el circuito. Generalmente se expresa con el número de circuitos análogos que puede controlar.

Fase. Representa el origen de una señal periódica.

Fault. Condición accidental de error en un periférico; generalmente determina su puesta fuera de servicio por parte del sistema operativo.

Fetch. Ciclo de toma de la memoria del código de la instrucción que deberá realizarse a continuación.

FF. Form Feed. Carácter de control (ver códigos ASCII) que produce el salto a una nueva página.

File jerárquico (Hierarchical file). File en el que algunos records dependen de otros según una estructura jerárquica en árbol.

File lógico (Logical file). Estructura de un file visto por el programa de aplicación. Puede ser completamente diferente de la estructura física.

Filtro (Filter). Componente que puede eliminar una magnitud dejando sin alteración las otras.

Firmware. Funciones de software memorizadas permanentemente, por ejemplo en ROM.

Flag. Un tipo cualquiera de indicador. Se utiliza en un cierto punto del programa para memorizar un estado o la producción de un suceso particular. Algunas veces se indica con los términos «código» o «indicador». En particular, algunos lenguajes, como por ejemplo el RPG, hacen mucho uso de los flags para determinar, por ejemplo, el tipo de record o para indicar el resultado de una comparación, etc. Generalmente, en estos lenguajes se utiliza el término código en lugar de flag, pero las funciones y la lógica de utilización son las mismas. Las operaciones activadas en función del estado de estos files se llaman «de ruptura de código».

Flat file. Matriz de datos de dos dimensiones.

Floppy. Diskette flexible.

FM. Frequency Modulation. Método de modulación de señales que consiste en variar la frecuencia de una señal portadora en función de los valores asumidos por la señal moduladora. Sinónimo de FSK.

Folding. Técnica de conversión de datos a la forma deseada partiendo de otra diferente. Por ejemplo, la transformación de las letras de minúsculas a mayúsculas.

Font. Juego de caracteres de una determinada dimensión y con características gráficas propias.

Foreground. Área de la pantalla ocupada por un dibujo o por un carácter (ver también background).

Formato (Format). Especifica la forma de salida o de entrada de datos. A veces se utiliza también para indicar la estructura de memorización en disco.

Formateado (Format). De un disco, operación que consiste en «preparar» el soporte según las especificaciones del sistema operativo particular.

Forward. Referido a la unidad de cinta o a un file secuencial, indica un desplazamiento hacia adelante en dirección al punto final. En las unidades de cinta se utiliza para indicar el enrollamiento de la cinta. En Pascal es una palabra reservada utilizada en las declaraciones de procedimientos particulares.

Frecuencia. Indica el número de veces por segundo en que una señal (periódica) asume el mismo valor; es el inverso del periodo.

FSK. Frequency Shift Key. Ver FM.

Full-duplex. Método de transmisión (ver Simplex) en que las informaciones se desplazan en dos sentidos al mismo tiempo.

Función (Function). Procedimiento que, de acuerdo con un algoritmo, restituye un valor dependiente de otros valores (parámetros). Se denominan intrínsecas las funciones ya previstas en un lenguaje (por ejemplo, las funciones matemáticas del Fortran).

Fusión (Merge). Procedimiento con el que dos vistas originan una única. Puede referirse también a dos programas, siempre con el mismo significado.

H

Habilitado (Enabled). Estado de un proceso que permite interrupciones. Es el término que indica la condición opuesta a deshabilitado.

Half duplex. Método de transmisión en los dos sentidos alternativamente (ver Duplex). Es el método más utilizado en las conexiones entre el vídeo, el teclado y la CPU.

Hard copy. Copia sobre papel de una salida.

Hardware. La parte física de un sistema de proceso.

Hard wired. Término con el que se indica un conjunto de conexiones eléctricas para la transferencia y el proceso de señales.

Hash total. Totalización de algunos campos de los records de un file con el fin de identificar un valor numérico (sin significado a los fines de los datos) a utilizar como elemento de control para que no se pierda ni se modifique ningún dato. Durante las transmisiones se utiliza una técnica análoga; en este caso, a veces se indica como «cuadratura».

Hashing. Técnica de direccionamiento que permite convertir la clave del record en un valor desde el cual se pide la dirección.

Header. En general indica una parte inicial que contiene la descripción o la clave de interpretación de las partes que siguen. El término se utiliza principalmente para indicar el record inicial de un file que contiene informaciones acerca de las que siguen.

Hertz (Hz). Unidad de medida de la frecuencia. Sus múltiplos son: kHz (kilo Hertz = 1000 Hz), MHz (Mega Hertz = 1.000.000 Hz), GHz (Giga Hertz = 1.000.000.000 Hz); no tiene submúltiplos.

Histograma (Histogram). Gráfico de barras. La longitud de

cada barra representa el valor de la variable en aquel punto.

Hit rate. Expresa el número de records que se espera que deben procesarse en un run. Normalmente es un valor porcentual referido a las dimensiones del file (en número de records).

Home address. Representa la dirección en la que debería colocarse lógicamente un dato. El término sólo se utiliza en los grandes sistemas para los cuales la gestión de los datos prevé un área, llamada de desbordamiento, en la que se depositan momentáneamente los records si en el momento de la introducción las áreas asignadas a ellos están ocupadas. También puede tener el significado de campo de direccionamiento (siempre referido a la memoria de masa).

Host. Ordenador primario en una instalación con más máquinas.

Housecleaning. Este término se emplea principalmente en el lenguaje Basic para indicar la operación de compactación del área de memoria reservada a las cadenas. Con esta operación, el ordenador elimina las cadenas que ya no se utilizan, reagrupa todas las áreas ocupadas en una zona contigua única y deja disponible el espacio restante para otros valores.

I

IC. Integrated Circuit. Circuito electrónico complejo, miniaturizado, realizado en una cápsula única.

Identificador (Identifier). Nombre dado a un programa o a una de sus partes.

Incremento (Increment). Valor que modifica un contador, por ejemplo el STEP en las instrucciones FOR... del Basic y DO... del Fortran.

Indentación (Indentation). Operación de inserción de un cierto número de espacios en blanco no significativos a la izquierda de una línea de programa con el fin de hacer el listado más legible. La indentación sólo es posible para algunos lenguajes, por ejemplo en Pascal o en algunas formas de Basic.

Index point. Referencia hardware utilizada generalmente para la sincronización.

Indexado. Que depende del valor de un índice.

Indicador. Ver Flag.

Índice. Valor utilizado para determinar la posición de un record o para identificar un determinado elemento de una tabla.

Indirecto (Indirect addressing). Método de direccionado que especifica una posición en la que hay contenida una

nueva dirección. El dato se encuentra en la dirección así especificada.

Inductancia. Elemento de circuito constituido por un cierto número de espiras. Se mide en μH (microhenrios).

Inicialización (Initialize). Implantación de los valores iniciales de flag, contadores, cadenas, etc. en un programa o en una subrutina.

Instrucción (Statement). Conjunto de palabras reservadas y datos que indican una particular acción pedida al ordenador. Cada programa está compuesto por una serie de instrucciones (ver también Palabra reservada y Palabra clave).

Instrucción compuesta. Una instrucción compleja no prevista en el lenguaje, obtenida reuniendo en un bloque cierto número de instrucciones elementales. Normalmente, el bloque debe empezar y terminar con determinadas palabras clave. La instrucción compuesta sólo está prevista en algunos lenguajes evolucionados.

Integridad (Integrity). Indica la ausencia de errores en un conjunto de datos.

Interactivo (Interactive). Es un programa que interacciona con el usuario (ver Conversacional). Método de desarrollo de un programa que actúa con la participación del usuario, por ejemplo para la validación de un dato acabado de introducir, y la eventual petición de correcciones. El opuesto es el batch, sistema que permite procesar los datos sólo al final de todas las instrucciones.

Intercambio (Exchange). Proceso en el que dos variables se intercambian el contenido.

Interfaz (Interface). Dispositivo que adapta las señales a las exigencias de circuitos diferentes. Por ejemplo, en las transmisiones entre diferentes ordenadores o entre un ordenador y los periféricos, las señales deben adaptarse tanto antes de la transmisión como después de la recepción.

Interno (Built-in). Identifica procedimientos que forman parte del sistema operativo.

Intérprete (Interpretive routine). Se refiere a un programa, o a rutinas particulares, que pueden interpretar y realizar un programa fuente. El intérprete es un programa que puede traducir un lenguaje simbólico en códigos realizables. A diferencia del Compilador, realiza la traducción de una instrucción cada vez, antes de su realización. La instrucción traducida no se memoriza, por lo que la operación debe realizarse tantas veces como cuantas son las veces que se utiliza la instrucción.

Interrupción (Interrupt). Suspensión del programa durante el desarrollo salvando todos los parámetros, para que la ejecución pueda reemprenderse después desde el mismo punto. Se dice que la interrupción está «recibida» cuando el sistema, en el estado habilitado, memoriza la petición. Se dice «servido» cuando después de suspendido el

programa en realización el sistema ha activado la rutina que realiza las tareas solicitadas por la interrupción.

Intersección (Intersection). Unión de dos o más grupos de datos obtenida tomando solamente los elementos comunes. Operación que, partiendo de dos conjuntos, crea un tercero agrupando todos los elementos comunes a los dos primeros.

Inverso (Inverse). Lo opuesto a un estado cualquiera. Con respecto a la pantalla vídeo, el término indica una escritura negra sobre fondo blanco o, en general, obtenida apagando los pixels que componen letras (normalmente es el opuesto, es decir, con pantalla apagada, las letras se obtienen activando algunas zonas).

Inverted file. Estructura particular de files que permite buscar informaciones no definidas con anterioridad. Generalmente, la función se obtiene construyendo oportunas listas de índices llamadas «inverted list».

I/O. Input/Output. Sigla que identifica cada operación general de introducción o de salida de datos.

IPS. Inches Per Second. Mide la cantidad de cinta (expresada en pulgadas) que puede procesarse (leída o escrita) por una unidad.

ISAM. Index Sequential Access Method. Método de acceso a los datos de un file por medio de índices.

Iterativo. Método de cálculo que permite buscar una solución por pasos sucesivos.

J

Jerarquía (Hierarchy). Estructura de elementos subdividida en diferentes niveles, con distintos grados de importancia o prioridad. Por ejemplo, la jerarquía de las operaciones establece las reglas de prioridad en la realización de las operaciones aritméticas. En los files o en las estructuras de datos, el término indica una subdivisión y los respectivos lazos de dependencia que permiten, dado un elemento de nivel más alto, pedir todos los demás, a nivel inferior, y correlacionados con este.

Joystick. Mecanismo constituido por una palanca con transductores de posición que permite, con sus desplazamientos, controlar la posición del cursor o de otro elemento en el vídeo.

Jump. Término con el que se indica una variación del proceso secuencial normal (salto).

K

k. Símbolo que indica kilo, en general un factor multiplicador por 1000. En los ordenadores, refiriéndose a las memorias, vale 1024.

L

Label. Término que indica, en muchos lenguajes simbólicos, una etiqueta de direccionamiento de una instrucción. También puede referirse al disco, y en este caso, indica su nombre. Conjunto de caracteres utilizados para identificar un record, un elemento o una línea de programa.

Latencia (Latency). Tiempo necesario para posicionar la cabeza de la unidad de disco sobre la superficie en rotación.

Layer. En los dibujos complejos, el conjunto de los elementos representados puede estar subdividido en subsistemas, cada uno indicado con el término layer, para proporcionar una vista lógica fraccionada que permite una composición del dibujo por grupos.

Layout. Dibujo que representa una entidad física normalmente a escala.

Leading. Parte inicial de un campo de datos u otro. Por ejemplo, los ceros antes de los valores numéricos o los espacios antes de los caracteres de una cadena.

Leaf Node. Este término se utiliza en la programación estructurada e indica un bloque que no tiene ramificaciones hacia abajo.

Lenguaje máquina (Machine language). El conjunto de códigos numéricos que el ordenador puede interpretar y realizar directamente, sin que sean necesarias las traducciones de los otros lenguajes.

LF. Line Feed. Carácter de control que produce el desplazamiento de la cabeza de escritura (carro) de una impresora o del cursor de la pantalla de vídeo desde la posición actual a la misma posición de la línea que sigue; así se tiene el desplazamiento de una línea sobre la misma columna.

LFU. Last Frequently Used. Algoritmo que permite sustituir los nuevos datos por los menos utilizados anteriormente (ver también LRU).

Librería (Library). Índice de los programas (fuente, objeto, etc.) generalmente utilizado por el sistema operativo.

Línea (Line). El término, referido a una salida, indica un conjunto de datos contiguos y terminados por CR. En la entrada es un grupo de datos considerados como un bloque único (por ejemplo, una cadena). En las transmisiones indica un medio cualquiera a través del cual pueden transmitirse las señales, por ejemplo los cables eléctricos o las fibras ópticas (para la transmisión de los datos en forma de señales luminosas).

Líneas ocultas (Hidden lines). Segmentos que deben anularse porque quedan cubiertos por otras partes, en los gráficos de objetos en tres dimensiones.

Listado (Listing). Copia sobre papel o sobre el video de las instrucciones que constituyen un programa.

Longitud fija (Fixed length). Normalmente indica un tipo particular de file en el que cada record debe tener la misma longitud que los otros, por ejemplo los files random en Basic.

Loop. Conjunto de instrucciones que pueden realizarse repetidamente hasta que es verdadera una determinada condición.

LRU. Last Recently Used. Algoritmo que permite sustituir los nuevos datos por los utilizados más recientemente (ver también LFU).

LSI. Large Scale Integration. Indica componentes de alto grado de integración. Normalmente se refiere a circuitos que comprenden un sistema completo.

LL

Llamada (Invoke). Activación de un procedimiento en uno de sus entry points.

M

M. Sigla que indica el factor de multiplicación Mega (1.000.000). Si se refiere a la memoria de un ordenador, vale 1.048.576 (ver también k, kilo).

Machine infinity. Indica el número más grande que puede representarse en el formato interno de un ordenador.

Macro. Grupo de comandos o de instrucciones vistos por el ordenador como una sola instrucción. Principalmente se utilizan en Assembler o en los paquetes software generalizados, como el tablero electrónico.

Macroinstrucción (Macroinstruction). Una instrucción fuente que genera una serie de acciones equivalentes a un cierto número de otras instrucciones.

Mainframe. Unidad central en una configuración de grandes dimensiones.

Mantisa (Mantissa). En la representación de los números con coma flotante (floating point), es la parte numérica que debe ser multiplicada por el exponente. Por ejemplo, el número 5600 escrito en la forma $5.6 \times E3$ tiene por mantisa el valor 5.6 y, por exponente, el valor 3.

Mapa (Map). Esquema o descripción del contenido de la memoria. En los gráficos se utiliza un mapa para indicar el estado de cada punto simple del video; en este caso toma el nombre de bit map.

Mapping. Referido a los files, indica la descripción de los plazos de dependencia entre los diversos records. Referido a la memoria, describe el uso de las varias zonas.

Mark. Estado 1 (alto) durante la transmisión de datos.

Máscara (Mask). Conjunto de caracteres utilizado para controlar la adquisición o la eliminación de algunos caracteres que pertenecen a otro conjunto. La máscara puede estar representada por otro carácter y, en este caso, se utiliza para seleccionar bits simples.

Master. Aparato DTE (ver) que controla el intercambio de datos en una conexión.

Matriz (Array). Un conjunto de elementos homogéneos en una o más dimensiones indicados con el mismo nombre simbólico e identificados con uno o más índices. Por ejemplo, es una matriz de dos dimensiones aquella en la que cada elemento está identificado por la fila y la columna a que pertenece.

Memoria asociativa (Associative storage). Memoria direccionada en base al contenido y no por medio de la dirección. El método permite un notable aumento de la velocidad de búsqueda (en base a los atributos del campo).

Memoria tampón (Buffer). Área de memoria utilizada como apoyo provisional de los datos. La memoria tampón se utiliza normalmente en las operaciones de I/O para compensar con un «pulmón» las diferentes velocidades entre los dispositivos que se intercambian los datos. Por ejemplo, durante una impresión, la presencia de una memoria tampón permite al ordenador enviar una cierta cantidad de datos y dedicarse a otras tareas mientras la impresora vacía dicha memoria y transfiere su contenido al papel.

Menú. Lista de las funciones previstas en un programa, con posibilidad de selección de la deseada.

Método de acceso (Access method). Técnica con la que es posible archivar un dato y mantenerlo disponible para el proceso. Los principales son: serie, directo (random), remoto, secuencial virtual (VSAM) e indexado jerárquico secuencial (HISAM). Estos dos últimos son característicos de los grandes sistemas.

Microordenador. Ordenador basado en circuitos integrados, a veces sobre un solo circuito.

Migración (Migration). Operación de transferencia de los datos a una zona en la que pueda aumentarse la velocidad de acceso.

Minifloppy. Normalmente indica los diskettes de 5" 1/4, con capacidades del orden de 160 kbytes. Recientemente también los de 3" 1/2, utilizados en algunos sistemas.

Miniordenador. Ordenador con capacidades superiores al

microordenador, con posibilidades de direccionamiento de memoria limitadas.

Modelo (Model). Los programas que simulan cierto fenómeno o determinado proceso utilizan un modelo para simular el comportamiento del sistema real.

Modelo conceptual (Conceptual model). Esquema de conjunto de un Data Base.

Modem. Modulador-demodulador. Aparato utilizado para modular y demodular las señales en las transmisiones a larga distancia.

Módulo (Module). Parte de programa lógicamente separada del resto que realiza funciones definidas y completamente contenidas.

Monitor. Sinónimo de pantalla de vídeo. En algunos casos, el término se utiliza para indicar la operación con la que un programa presenta el estado de las variables mientras continúa con el proceso.

MSI. Medium Scale Integration. Tecnología particular que permite reunir en un elemento único un cierto número de componentes (inferior al tipo LSI).

Multiplexador. Aparato que permite el coloquio de varias unidades utilizando un reducido número de conexiones. Los tipos utilizados son dos: de división de tiempo y de división de frecuencia.

Multiprocesador (Multiprocessor). Ordenador que utiliza varias unidades al mismo tiempo.

Multipunto (Multipoint). Tipo de configuración de red en el que un DTE amo tiene el gobierno de varios DTE esclavos.

N

Normalización (Normalize). Alineado de las cifras que componen un número para obtenerlo en la forma requerida por las reglas aritméticas.

Notación (Notation). Forma de representación de los datos. Las notaciones utilizadas en los ordenadores son la binaria, la octal y la hexadecimal.

Núcleo (Core). Elemento de hardware de memoria basado en la magnetización de núcleos de material ferromagnético.

Nudo (Node). Referido a las redes, indica el punto terminal de una gama o la conexión de dos o más derivaciones. Punto de derivación de un diagrama de flujo. El término se emplea principalmente en la programación estructurada e indica una desviación de un diagrama en árbol.

Null. Conjunto vacío. Por ejemplo, en Basic, una cadena sin caracteres. Palabra en blanco o que en general no comporta ninguna acción, interpuesta entre palabras que contienen los datos en las transmisiones sincronas.

O

Octal. Sistema de numeración posicional que utiliza el valor 8 como base.

Off-line. Aparato bajo el control directo del ordenador que controla el sistema. Referido a un proceso, indica su desarrollo en el flujo principal.

Offset. Valor que indica el posicionado de un dato a partir de un origen. Por ejemplo, en un file puede ser el número de bytes que indica la posición del dato a partir del principio del record. Referido a la memoria, indica un valor, constante, que se debe restar o sumar a la dirección para conocer la posición real de un dato. En términos eléctricos indica una tensión constante a partir de la cual se aplica la señal.

Omisión (Default). Valor opcional asumido por el sistema a falta de otra indicación.

On-Line. Indica un sistema en el que los datos pueden introducirse o tomarse directamente. Es el opuesto al off-line, utilizado para indicar datos solamente accesibles después de las oportunas acciones, por ejemplo la copia en cinta de archivos históricos.

Operación (Operation). Una acción que aplicada a una combinación de elementos conocidos, genera un nuevo elemento perteneciente al mismo conjunto.

Operador (Operator). Símbolo particular que indica la operación a realizar sobre los operandos. Los operadores principales son los aritméticos, que realizan las operaciones aritméticas normales y las lógicas.

Operando (Operand). Elemento sobre el que debe realizarse una operación.

Orden (Order). Indica la lógica utilizada para formar una lista particular de datos. Si se refiere a un bit, indica la posición en el interior de la palabra.

Ordenado de burbuja (Bubble Sort). Técnica que realiza un sort (ordenado) en una tabla y que consiste en seleccionar una parte de los elementos de la tabla, intercambiando sus posiciones para obtener el ordenado. El procedimiento se repite hasta obtener el ordenado de toda la tabla. El nombre de «burbuja» se deriva del movimiento de los elementos hacia la parte superior de la lista. Una versión particular de Bubble Sort, más rápido si ya existe un ordenamiento parcial, se tiene considerando un valor hacia la parte superior de la lista y uno hacia el final. Toma el nombre de «Cocktail Shaker Sort».

Oscilador (Oscillator). Circuito que puede generar una señal periódica (ver Reloj y Cristal).

P

Pad. Pequeña área de un circuito impreso sobre la que se

sueldan las conexiones de los componentes.

Padding. Relleno de un bloque con datos dummy, por ejemplo ceros o espacios.

Página (Page). Área de memoria constituida generalmente por 1024 posiciones. En algunos sistemas se distinguen la página cero y la página en curso. La página cero contiene informaciones de enlace utilizadas por el sistema; la página en curso es la zona de memoria que contiene la instrucción en curso de realización. Esta subdivisión en algunos sistemas está determinada a fines de los direccionamientos de memoria.

Página activa (Active page). En algunos sistemas, la unidad puede utilizar diferentes páginas de memoria; la página activa indica cuál de ellas contiene informaciones. Generalmente es diferente de la presentada.

Paging. Técnica de segmentación, utilizada por los sistemas operativos más avanzados, que consiste en dividir los programas o los datos de página normalmente residentes en disco y cargados en memoria, sólo cuando se producen. De esta manera, la memoria aparece mucho más amplia de lo que es en la realidad (la misma zona está compartida, en tiempos diferentes, por grupos distintos de datos).

Palabra (Word). Conjunto de bits en un número igual al que puede contenerse en una memoria. Las longitudes más utilizadas son 8, 16, 32 bits.

Palabra clave (Keyword). Una palabra predefinida y reservada, en el lenguaje de programación utilizado, para una determinada finalidad.

Palabra reservada (Reserved word). Término utilizado en un lenguaje o en un comando para activar una determinada función. En general no puede utilizarse para otras finalidades, por ejemplo, como nombre de variable.

Parámetro (Parameter). Nombre simbólico utilizado para la transferencia de valores, como argumento de una función, en la llamada de una subrutina o entre procedimientos.

Paridad (Parity). Método de validación de datos en las transmisiones. Consiste en verificar el número de estados ON enviando, junto con el dato, un bit que indica si este número es par o impar. «Paridad par» indica la lógica según la cual el bit de paridad es activo si el número de bits en ON es par; en paridad impar, el bit es activado cuando este número es impar.

Paso. Operación de transferencia de parámetros entre un programa (o subrutina) llamador y la subrutina (o función) llamada.

Pattern. Indica una serie de elementos conectados lógicamente. En Pascal se utiliza para indicar una cadena que deberá compararse con otra para determinar si está contenida en ella. En telecomunicación indica una serie de datos.

PCB. Printed Circuit Board. Circuito impreso, constituido por un soporte aislante, por ejemplo fibra de vidrio, sobre el que se ha representado, por medio de procesos fotográficos, el conjunto de conexiones a realizar.

PEP. Parametric Element Processor. Lenguaje de alto nivel utilizado para resolver problemas de gráficos tridimensionales.

Periférico (Peripheral device). Cualquier dispositivo utilizado por el ordenador para comunicar con el mundo exterior.

Periodo. Referido a una señal periódica, indica su duración.

Peso. Valor multiplicador de una cifra que depende de su posición en un número escrito según un sistema de numeración posicional. Por ejemplo, la cifra 7 en el número 275 tiene un peso de 10.

Pista (Track). División circular del disco, subdividida en sectores.

Pixel. Un punto de la pantalla en modo gráfico. También puede referirse al bit particular del área de memoria que contiene la información acerca del estado del punto de pantalla correspondiente.

Pluma óptica (Light pen). Dispositivo sensible a la luz utilizado para identificar un punto de la pantalla en base a su luminosidad.

Polling. Fase de transmisión de datos en la que el amo invita a un esclavo a transmitir.

Posición (Location). Identifica una memoria o un espacio en el disco. En general indica una zona cualquiera en la que pueda memorizarse un dato. También es una posición identificable con un valor numérico. Por ejemplo, un record en el interior de un file o un carácter en una cadena.

Posición (Position). Posición identificable con un valor numérico. Por ejemplo, un record en el interior de un file o un carácter en una cadena.

Postprocessor. Programa que puede interpretar datos a la salida de otro software para convertirlos en un formato adecuado a usos sucesivos, por ejemplo para generar dibujos con el plotter.

Precisión (Precision). Indica la «calidad» de la máquina. En los ordenadores puede estar indicada por la diferencia entre el valor verdadero de un número y el que permite gestionar la máquina. Por ejemplo, en una máquina que utiliza 9 cifras significativas (para los números reales), la precisión es de aproximadamente 1/1.000.000.000.

Predecesor. Elemento que precede en un ordenado lógico predefinido.

Preprocessing. Método de proceso de los datos para

hacerlos homogéneos con las especificaciones de un software particular.

Procedimiento (Procedure). Conjunto de los programas que realizan una tarea específica. En Pascal tiene un significado diferente. Indica una parte del programa que puede enviarse a ejecución; por tanto, tiene el significado de subrutina.

Proceso distribuido (Distributed Processing). Proceso de datos obtenido, simultáneamente, en diversos sistemas conectados en red.

Programa de aplicación (Application Program). Un programa escrito para la solución de un problema específico.

Programación estructurada (Structured programming). Método lógico de desarrollo del software que consiste en subdividir el problema en problemas menores, resolviendo cada uno de ellos con un módulo conceptualmente separado de los otros.

Prompt. Mensaje o símbolo utilizado por el ordenador para guiar al operador.

Protegida (Protect). Área de memoria de acceso controlado. Puede referirse a un diskette y, en este caso, indica la presencia o no del bloque hardware de la escritura (este bloque normalmente está realizado cubriendo, o en algunos casos descubriendo, una pista en el disco).

Protocolo (Protocol). Conjunto de reglas que controlan el intercambio de mensajes entre dos unidades.

Puck. Aparato manual utilizado para introducir coordenadas.

Puerta (Port). Punto de acceso para la entrada o la salida de los datos. Las puertas pueden ser digitales, si permiten la transferencia de señales digitales (ON/OFF), o analógicas. En este caso deben ir provistas de convertidor analógico/digital o viceversa.

Puntero (Pointer). Variable que con su valor indica un elemento particular de un conjunto. Puede referirse ya a un elemento de una matriz, a un record o a un byte del interior del record.

Punto direccionable (Addressable point). En los ordenadores gráficos, la pantalla está subdividida en un elevado número de puntos. Un dibujo se constituye activando algunos de ellos. En general, no toda la pantalla puede gestionarse; las zonas activables se llaman «puntos direccionables».

R

Raíz (Root). En términos matemáticos representa una

operación particular. Pero la programación estructurada es el nudo de base de un diagrama de árbol.

RAM. Random Access Memory. Indica memorias de acceso directo (ver Acceso selectivo).

Random. Método de direccionamiento en base a la posición; sinónimo de acceso directo.

Randomización. Algoritmo para la generación de números aleatorios.

Range. Límites entre los cuales está comprendido el valor de un determinado elemento. Por ejemplo, el range de los números enteros en una máquina de 8 bits va de aproximadamente -32.000 a aproximadamente +32.000.

Raster display. Pantalla de vídeo en la que toda su superficie es explorada a intervalos regulares, llamados «refresh rate».

Raster scan. Método de formación de la imagen televisiva basado en la excitación de algunos puntos de la pantalla mientras la explora el haz. Método de generación de un dibujo que consiste en su formación línea por línea explorando toda la pantalla. Es un método similar al utilizado para la formación de la imagen televisiva.

Read-only. Dispositivo (generalmente memoria), utilizable sólo en lectura, o sea que no permite modificaciones.

Real. Variable o constante que pertenece al conjunto de los números reales.

Record. Conjunto de informaciones correlacionadas tratadas como una entidad única.

Recursivo (Recursive). Procedimiento en el que cada paso utiliza los resultados del anterior. Por ejemplo, la técnica que permite obtener la solución de una ecuación con aproximaciones sucesivas, o particulares tipos de rutinas (también llamadas «de reentrada») que pueden llamarse a sí mismas.

Red. Dos o más ordenadores interconectados.

Redondeo (Round). Método matemático que consiste en eliminar algunas cifras de un número teniendo en cuenta las más significativas. Permite aproximar un valor real al número más cercano que el procesador puede gestionar.

Refrescado (Refresh). Método de mantenimiento de la imagen vídeo que consiste en volverla a dibujar a intervalos de tiempos regulares, antes de que desaparezca.

RGB. Sigla utilizada para indicar un monitor en colores basado en la técnica aditiva (Rojo, Verde, Azul).

Relacional. Data Base particular constituido en base a relaciones. Su gestión prevé el uso de software que puede procesar los elementos que lo constituyen de manera que genere nuevas relaciones, confiriendo una gran flexibilidad de empleo al sistema.

Reloj (Clock). Señal, generalmente de forma cuadrada, sobre la que se sincronizan los componentes del sistema. Componente que puede generar una señal periódica utilizada para la sincronización. Cada señal es un impulso (clock pulse) y se caracteriza por una amplitud y por un periodo. La amplitud es el valor máximo que alcanza la señal en el estado ON, el periodo es su duración en segundos. A veces como característica se utiliza la frecuencia (medida en Hz o en MHz), cuyo valor numérico es inverso al periodo y que representa el número de impulsos emitidos en un segundo.

Repaint. Operación de reconstrucción de un dibujo para incluirle las últimas modificaciones.

Resistor. Componente de circuito que ofrece una resistencia al paso de corriente. Se mide en ohmios y sus múltiplos ($k = \times 1.000$, $M = \times 1.000.000$).

Resolución (Resolution). Representa la cantidad mínima que puede detectarse. En los gráficos es el número de líneas que pueden distinguirse en la unidad de longitud. En los transductores es la cantidad mínima de la magnitud examinada que da lugar a una señal utilizable.

ROM. Read-Only Memory. Memoria de sólo lectura (ver Read-only).

Router. Software que puede determinar automáticamente las interconexiones de los componentes en un circuito impreso.

RS232. Interfaz de comunicación más utilizado en las conexiones para la transmisión de datos entre ordenadores o entre ordenador y periférico.

RTL. Resistor Transistor Logic. Circuito lógico obtenido combinado resistencias y transistores.

Rubber Banding. Técnica de generación de una línea que consiste en fijar uno de sus extremos y dibujar el otro de acuerdo con los desplazamientos de un periférico de entrada, por ejemplo el joystick.

Ruido (Noise). Señal de origen indeterminado que se superpone a la señal útil, generando parásitos.

Run Time. Periodo durante el cual un programa está activado. En algunos sistemas operativos o lenguajes, el término indica un módulo particular que contiene las informaciones necesarias para utilizar un programa.

Rutina (Routine). Parte de programa que contiene instrucciones utilizables por otras partes diferentes del mismo por medio de un salto con retorno.

S

Scissor. Software para la subdivisión de gráficos en partes

más pequeñas contenidas en las dimensiones de la pantalla de vídeo.

Scroll. Movimiento horizontal o vertical de la imagen en la pantalla de vídeo para dejar sitio a nuevos datos. En sistemas más complejos, con una adecuada memoria de vídeo, el scroll puede producirse en los dos sentidos (es decir, pueden reclamarse líneas o columnas ya trasladadas).

Sector. Cantidad mínima direccionable en la memoria de masa.

Secuencial (Index-Sequential). Tipo particular de file en el que los records se memorizan en secuencia ascendente según el contenido de un campo particular de la clave.

Segmento (Segment). Indica una zona de memoria definida como extensión, por ejemplo 64 kbytes.

Selección (Selection). Fase de transmisión de datos en la que se invita a un esclavo a recibir.

Semántica. Conjunto de reglas que describen un lenguaje de programación.

Serie (Series). Expresión que contiene un conjunto de elementos, cada uno de los cuales se genera con precisas reglas matemáticas.

Set. Conjunto de símbolos reconocidos en un lenguaje o por un procesador.

Side Effect. Normalmente indica un error generado a causa de volverse a cargar en partes del programa diferentes de aquella en que se manifiesta el error.

Simplex. Método de transmisión de datos que permite una sola vía de comunicación, por ejemplo sólo del ordenador al periférico, o viceversa.

Single Step. Realización de un programa con pasos simples. Es una técnica utilizada para la búsqueda de los errores. Esta posibilidad no está presente en todas las máquinas.

Sintaxis (Syntax). Reglas que describen cómo utilizar las instrucciones de un determinado lenguaje. Pueden estar representadas sintéticamente con los diagramas sintácticos.

Sistema operativo (Operating system). Software que controla la realización de los programas de aplicación, incluidos eventuales intérpretes o compiladores y las rutinas de gestión de los periféricos.

Slave. Aparato DTE (ver) que forma parte de una red pero sin funciones de control; por ejemplo, un terminal de vídeo (ver también Master).

Software. Programas, rutinas, y en general todo lo que no es un objeto físico.

Step-wise. Técnica de desarrollo de un programa que consiste en la obtención del resultado final a través de sucesivos afinados con crecientes grado de detalle.

Subcadena (Substring). Serie de caracteres tomados por extracción de una cadena.

Suceso. Acción generalmente externa que puede producir una interrupción de programa (interrupt) para activar determinadas rutinas. En algunos ordenadores no hay previstas teclas funcionales, gestionadas bajo interrupción, cuya activación genera un suceso similar. El otro modo de gestión de las teclas funcionales es con programa, en este caso no se tiene un suceso de interrupción, sino una interrogación continua, por parte del programa, del estado de las teclas.

Superposición (Overlay). Utilización de la misma área de memoria por parte de varios programas en tiempos sucesivos. En los sistemas más pequeños, la superposición debe ser gestionada por el programa de aplicación; en los más evolucionados está regulada por el sistema operativo en función de la actividad en curso.

T

Tambor (Drum). Referido a los plotters, indica un tambor sobre el cual se desplaza el papel. Puede referirse a memorias de masa particulares.

Teclado (Keyboard). Es el conjunto de las teclas, similares a las de una máquina de escribir, que sirven para la introducción de los datos en el ordenador.

Terminal. Dispositivo de I/O en una red de telecomunicaciones.

Tiempo compartido (Time sharing). Indica la utilización de la misma memoria por parte de varios programas.

Tiempo real (Real time). Indica una aplicación en la que las acciones que siguen a las introducciones se realizan inmediatamente.

Track-ball. Dispositivo de entrada constituido por una esfera, girando la cual se tiene el desplazamiento correspondiente del cursor. Su funcionamiento es análogo al del joystick.

Transacción (Transaction). Operación que genera la creación o el borrado de un record.

TRC (CRT, Cathode Ray Tube). Tubo de rayos catódicos. Componente que puede presentar los datos en forma visible, aprovechando la propiedad que tienen algunos fósforos de ser excitados por un haz de electrones. Es el componente principal del monitor.

Tres estados (Three state). Dispositivo electrónico digital particular que puede ponerse en un estado de alta impedancia (además de los estados 0 y 1).

Troughput. Velocidad con que se realiza un proceso.

TTY. Abreviación de teleinscriptora.

U

UART (Universal Asynchronous Receiver Transmitter). Dispositivo programable para la gestión de las funciones I/O.

Ubicación (Allocate). Indica la operación con la que se asigna un recurso (por ejemplo un periférico o un área de memoria) a un usuario específico o a un programa (en los sistemas que prevén la multiprogramación).

V

Validación (Validate). Operación de comprobación de datos, normalmente realizada en los programas de aplicación.

Velocidad de transferencia (Transfer Rate). Velocidad con la que se transfieren los datos entre un dispositivo y la

unidad central; normalmente se expresa en miles de caracteres por segundo.

Ventana (Window). Zona rectangular de la pantalla seleccionada por el operador. En los gráficos, y en algunos casos también en modo texto, indica una zona particular de la pantalla, gestionada de manera independiente del resto, como si fuese un vídeo separado. En Pascal tiene el significado de conjunto de identificadores y se indica con el término window diagram.

W

Walking-one. Técnica de comprobación de las memorias.

Z

Zócalo (Socket). Alojamiento del material aislante que soporta los componentes y evita así su soldadura en el circuito impreso.

ZOOM. Capacidad de presentar parte de un dibujo en una escala ampliada.



Indice de materias

Volumen 1

Pág.	13	EL ORDENADOR, UN CELEBRE DESCONOCIDO
»	15	Los datos
»	22	De la calculadora al ordenador: el programa
»	22	Pasado y presente
»	28	Aplicación y evolución de los ordenadores
»	33	LA LOGICA DEL ORDENADOR
»	33	Ordenadores analógicos
»	33	Ordenadores digitales
»	37	Dónde y cómo utilizar un ordenador
»	38	Clasificación de los ordenadores
»	43	Redes de ordenadores
»	45	LOS SISTEMAS DE NUMERACION
»	45	Los impulsos
»	46	Conmutación
»	47	Las bases de numeración
»	50	Notación binaria
»	52	Notación hexadecimal
»	55	Notación octal
»	56	Operaciones en notación binaria
»	60	Suma binaria
»	61	Diferencia binaria
»	72	ELEMENTOS DE LOGICA
»	72	Los operadores
»	73	Los operadores lógicos
»	79	Lógica cableada
»	80	Circuitos lógicos
»	80	Operador NOT
»	81	Operador AND
»	89	Operador OR
»	90	Operador XOR
»	90	Otros tipos de inversor
»	97	Circuitos integrados
»	97	Circuitos compuestos
»	99	Aplicaciones
»	100	Otros tipos de circuitos integrados
»	107	LOS CODIGOS DE TRANSMISION DE DATOS
»	107	Bit y byte
»	110	El código ASCII
»	111	Códigos transparentes
»	114	Símbolos y números
»	114	Letras
»	115	Aplicaciones
»	117	Transmisión de los códigos ASCII
»	121	Control de recepción
»	121	Códigos de seguridad
»	125	SISTEMAS DE MICROORDENADOR
»	125	Estructura básica
»	126	Estructura de la CPU
»	126	Funcionamiento de la CPU
»	129	Fases de una instrucción de salto
»	132	Interrupciones de programa
»	133	Lectura de los datos

Pág.	139	Memorias
»	140	Periféricos
»	142	Teclado
»	142	Impresora
»	142	Unidades de disco
»	146	Las instrucciones en la CPU
»	147	Instrucciones de transferencia de datos
»	148	Instrucciones aritméticas y lógicas
»	148	Instrucciones de salto
»	148	Instrucciones I/O (Entrada/Salida)
»	148	Instrucciones especiales
»	154	LA PROGRAMACION
»	154	Los lenguajes de programación
»	154	Lenguajes de alto nivel
»	156	Compilación
»	156	Basic
»	156	Fortran
»	156	Cobol
»	157	ANALISIS Y DIAGRAMAS DE FLUJO
»	157	Planeamiento de un programa
»	158	Análisis
»	162	Síntesis
»	168	Diagramación
»	168	Símbolos de uso corriente
»	170	Símbolos gráficos menos frecuentes
»	172	Ejemplo de aplicación
»	174	Los bucles
»	175	Bucle entre límites explícitos
»	175	Bucle entre límites parametrizados
»	176	Bucle con indicador de fin de sondeo
»	177	Bucle con ruptura de código
»	186	Bucles con paso distinto de 1
»	190	Salida forzada de los bucles
»	193	Aplicaciones de los diagramas de flujo
»	200	Aplicaciones científicas
»	203	Aplicaciones del método iterativo
»	209	Programación estructurada
»	220	ARCHIVO DE DATOS
»	220	Memorias de masa
»	220	Cintas magnéticas
»	227	Discos magnéticos
»	229	Tambores magnéticos
»	229	Floppy
»	231	Los ficheros
»	236	Organización de archivos en cinta magnética
»	238	Organización de archivos en disco magnético
»	238	Organización secuencial
»	238	Organización secuencial concatenada
»	240	Organización secuencial con índice
»	242	Organización con acceso directo
»	246	Organización de archivos en los microordenadores y ordenadores personales
»	248	Files directos (random)
»	248	Files secuenciales
»	248	Files con índices
»	254	Gestión de un file con índice (sistema ISAM)
»	255	Operaciones de acceso a los datos
»	256	Estructura general del file ISAM
»	256	La estructura B-tree
»	258	La estructura ISAM de árbol

Pág.	258	Número de files abiertos y files buffering
»	259	Direccionamiento directo
»	259	Subdivisión
»	260	Multiplicación
»	260	Multiplicación por 11
»	260	Cuadratura
»	261	Ejemplo de aplicación: gestión del balance familiar
»	261	El plan de cuentas
»	262	Módulo para el cálculo progresivo
»	269	Módulo de control de introducciones
»	271	La búsqueda de los datos en los archivos
»	273	Búsqueda en serie
»	275	Búsqueda binaria
»	275	Ordenación de los datos
»	283	Compactación de datos
»	283	Bancos de datos en los grandes sistemas
»	285	Base de datos relacional

Volumen 2

»	289	EL PLANTEAMIENTO DE LOS PROGRAMAS
»	289	Definición de las salidas necesarias
»	290	Comprobación de los datos de entrada y los algoritmos de cálculo
»	291	Elección de la máquina
»	291	Sistemas operativos
»	292	Las funciones del sistema operativo
»	295	Gestión de files en floppy
»	296	Formateado
»	298	Directorio
»	298	Tabla de los extnt
»	298	Referencia a los files
»	299	Funciones de SO relacionadas con los files
»	311	Funciones de paso al ambiente Basic
»	314	Criterios de evaluación de un sistema operativo
»	314	Subdivisión de un programa
»	316	Transferencia de parámetros
»	317	EL LENGUAJE DE LA PROGRAMACION BASIC
»	320	Generalidades
»	320	Modalidades operativas
»	322	Estructura de las instrucciones Basic
»	322	Funciones de control
»	323	Uso inmediato del Basic
»	323	Simbología, significado y prioridad de los operadores
»	323	Operadores aritméticos
»	326	Operadores relacionales
»	326	Operadores lógicos
»	332	Operadores funcionales
»	337	Operador división entera
»	337	Operador módulo
»	338	Constantes y variables
»	338	Constantes numéricas
»	339	Precisión
»	340	Variables numéricas
»	343	Conversiones entre los distintos tipos de variables
»	343	Redondeos
»	345	Redondeo en las asignaciones
»	345	Redondeo en los cálculos
»	347	Redondeo en las operaciones lógicas
»	347	Variables estructuradas
»	347	Matrices «array»
»	348	Cadenas

Pág.	352	Subcadena
»	353	Los comandos
»	353	Comandos para la escritura y preservación de programas
»	363	Comandos para la cinta magnética
»	363	Impresión y listado de programas
»	363	Activación e interrupción de programas
»	364	Encadenamiento
»	364	Las instrucciones
»	366	Inicio y final de los programas
»	368	Declaración del tipo de las variables
»	370	Instrucciones de asignación
»	370	LET
»	372	DATA, READ, RESTORE
»	382	LSET y RSET
»	382	SPACES(N)
»	384	SPC(N)
»	385	SWAP
»	385	Bucles
»	386	FOR... NEXT...
»	393	WHILE... WEND
»	396	Instrucciones de salto
»	397	GOTO...
»	397	ON... GOTO...
»	401	Instrucciones condicionales
»	409	Uso de las variables reales en las sentencias condicionales
»	411	Llamada a la subrutina y encadenamiento de programas
»	413	Ejemplo aplicativo: ¿diesel o gasolina?
»	417	Errores y su diagnóstico
»	418	Diagnóstico para la introducción de datos
»	434	Las funciones
»	435	Definición de una función por puntos
»	437	Interpolación y extrapolación
»	439	Representación analítica de las funciones
»	440	Las funciones en el Basic
»	440	Funciones numéricas
»	440	ABS(R)
»	441	CDBL(R)
»	442	CINT(R)
»	442	CSNG(R)
»	442	FIX(R)
»	443	INT(R)
»	443	SNG(R)
»	443	SQR(R)
»	444	RND(R)
»	447	ATN(Y)
»	447	COS(R)
»	447	SIN(R)
»	447	TAN(R)
»	448	Aplicaciones de las funciones trigonométricas
»	449	EXP(R)
»	450	LOG(R)
»	450	Funciones de cadena
»	450	ASC(A\$)
»	450	CHR\$(N)
»	452	CVI(A\$)
»	454	CVS(A\$)
»	454	CVD(A\$)
»	454	HEX\$(N)
»	454	INKEY\$
»	455	INPUT\$(N)
»	455	INSTR(N,A\$,B\$)
»	457	LEFT\$(A\$,N)
»	458	LEN(A\$)
»	458	MID\$(A\$,NS,NC)
»	460	MKIS(N)

Pág.	460	MK\$(R)
»	460	MK\$(D)
»	460	OCT\$(N)
»	460	RIGHT\$(A\$,N)
»	465	STR\$(R)
»	468	STRING\$(N,K)
»	469	VAL(A\$)
»	469	Funciones especiales
»	469	EOF(N)
»	470	FRE(A)
»	470	INP(N)
»	470	LOC(N)
»	471	LPOS(N)
»	472	OUT N,M
»	472	PEEK(N)
»	472	POKE N,M
»	473	POS(N)
»	473	Funciones definidas por el usuario
»	481	Funciones numéricas: solución de ecuaciones de segundo grado
»	485	Funciones de cadena: control de mayúsculas y minúsculas
»	488	Aplicaciones concretas
»	491	Ejemplo de aplicación: calefacción de ambientes
»	496	Análisis general
»	501	Las subrutinas
»	505	El programa
»	520	Las matrices
»	521	Memorización de los datos en una matriz
»	521	Matrices mono y dimensional
»	522	Instrucciones de asignación para las matrices
»	529	Aplicaciones de las matrices
»	531	Desarrollo de cálculos con un número cualquiera de cifras significativas
»	537	Aplicaciones a la estadística
»	547	Las funciones de entrada y de salida de datos
»	548	Funciones de introducción de datos
»	548	INPUT
»	550	INKEY\$
»	554	INPUT\$(N)
»	555	LINE INPUT
»	555	Funciones de emisión de datos
»	556	LPRINT
»	557	TAB(N)
»	557	LPRINT USING
»	560	Estructuración de las impresoras
»	560	Generación de listados no parametrizados
»	566	Generación de listados parametrizados

Volumen 3

»	577	Gestión de la impresora
»	577	Estructura y características de una impresora de agujas
»	581	Ejemplo de impresión
»	588	Generación de diagramas de barras
»	594	Gestión de la consola de vídeo
»	594	Estructura y características de una consola de vídeo
»	595	CONVERSACIONAL
»	596	PAGE
»	596	SCROLL
»	595	PROTECT
»	596	Gestión del cursor
»	596	La máscara vídeo
»	605	Las teclas funcionales
»	616	El Vídeo File Editor Olivetti
»	617	Las funciones de edición y de búsqueda de las cadenas
»	617	Los menús

Pág.	630	Generación de histogramas
»	637	Gestión de los archivos
»	637	Funciones de acceso a los files secuenciales
»	638	INPUT
»	638	LINE INPUT
»	638	WRITE
»	639	PRINT y PRINT USING
»	639	EOF(N)
»	639	LOC(N)
»	639	KILL
»	639	NAME
»	639	Adición de datos a un file secuencial
»	642	Funciones de acceso a los files directos
»	642	OPEN
»	643	FIELD
»	643	CLOSE
»	643	PUT
»	646	GET
»	646	LOC(N)
»	647	Formatos y codificación de los datos
»	654	Selección de los datos
»	659	Comandos y funciones particulares
»	661	Los archivos multivolumen
»	662	Gestión de los archivos en los grandes sistemas
»	662	La gestión de un almacén
»	663	Revaluación de las existencias e inventario
»	665	Desglose arbóreo
»	665	Simulación
»	665	Enlace con otros procedimientos
»	666	Ejemplo de aplicación: la gestión de una nómina
»	666	Apertura del file (subrutina 1000)
»	670	Menú (subrutina 2000)
»	670	Introducción (subrutina 3000)
»	670	Actualización (subrutina 4000)
»	672	Búsqueda (subrutina 5000)
»	673	Impresión (subrutina 6000)
»	673	Funciones de impresión particulares
»	675	El ordenamiento de los datos
»	676	Ordenamiento en memoria
»	680	Ordenamiento en disco
»	687	El programa de utilidad OLISORT
»	689	Prestaciones
»	692	Modalidad de ejecución
»	695	Desarrollo de un procedimiento de gestión de datos
»	695	La fase de introducción
»	695	Preparación de la máscara vídeo
»	699	Presentación de la máscara vídeo
»	703	Introducción de los datos
»	708	La fase de memorización en el disco
»	709	La función de escritura
»	712	La función de lectura
»	712	El programa principal
»	717	La compilación
»	717	La fase de preparación
»	717	BASCOM.COM
»	717	BRUN.COM
»	719	BASLIB.REL
»	719	OBSLIB.REL
»	719	BCLOAD
»	719	L80.COM
»	719	La fase de compilación
»	720	Objeto
»	720	Lista
»	720	Fuente
»	722	Funciones particulares del Compilador

Pág.	722	Especificación de las convenciones
»	725	Funciones de trampa de los errores
»	726	Códigos especiales
»	726	La fase de encadenamiento
»	733	Instrucciones particulares y compendio del Basic 80
»	733	Instrucciones de uso general
»	733	CALL
»	736	COMMON
»	736	USR
»	738	STOP
»	738	Instrucciones orientadas a la gráfica
»	740	Instrucciones particulares
»	740	Generación de sonidos
»	740	Joystick y paddle
»	741	El lápiz óptico
»	746	Los dialectos del Basic
»	746	Gestión de las cadenas
»	747	Gestión de los archivos en disco
»	753	Gestión de las funciones I/O
»	755	Gestión de los periféricos: la adaptación
»	760	Transmisión y recepción de datos
»	769	EL SOFTWARE
»	770	La arquitectura del software
»	772	La modelización
»	773	Modelo macroscópico
»	775	Modelo microscópico
»	776	Estructuración de los programas
»	776	La técnica de desarrollo top-down
»	784	Los errores del software
»	784	Aprobación del software
»	786	Métodos de aceptación
»	787	Bottom-up
»	788	Top-down
»	788	Big-Bang
»	788	Sandwich
»	788	Gráficos causa-efecto
»	789	Eliminación de los errores
»	791	La seguridad del software
»	793	Los programas de aplicación generalizados
»	793	El tablero electrónico
»	798	El estado introducción
»	799	El estado comandos
»	800	FORMAT
»	801	GLOBAL
»	801	TITLES
»	813	Las funciones del tablero electrónico
»	813	Funciones matemáticas
»	814	Funciones numéricas específicas
»	814	AVERAGE (n1, n2, ...)
»	817	COUNT (NS..NE)
»	817	SUM (NS..NE)
»	818	MAX (NS..NE), MIN (NS..NE)
»	818	NPV (S,NS..NE)
»	818	Funciones de búsqueda
»	818	LOOKUP (M,NS..NE)
»	820	Funciones lógicas
»	820	AND (lista de expresiones)
»	820	IF (condición, verdadero, falso)
»	822	NOT (condición)
»	822	ISNA (celda)
»	822	ISERROR (celda)
»	823	OR (condiciones)
»	823	Funciones privadas de argumentos

Pag.	823	PI
»	823	ERROR
»	823	FALSE/TRUE
»	823	NA
»	825	Ejemplos de aplicación del tablero electrónico
»	825	Distribución de los gastos comunitarios
»	832	Gestión de un almacén
»	835	Análisis de las variaciones (WHAT IF)
»	843	Las funciones para el cálculo recurrente
»	843	ITERCNT ()
»	843	DELTA ()
»	845	Evolución del tablero electrónico
»	847	Los programas de gestión de los archivos
»	847	Creación de los files y definición de los campos
»	849	Introducción y actualización de los datos
»	849	Búsqueda (Find)
»	851	Ordenación (Sort)
»	851	Elaboración de los datos
»	858	Impresión de los tabulados
»	858	Tratamiento de textos

Volumen 4

»	865	EL LENGUAJE ASSEMBLER
»	866	Estructura y funcionamiento del microprocesador
»	870	Los registros
»	871	Funcionamiento de la unidad central de proceso
»	876	Arquitectura interna del microprocesador
»	877	Unidad lógico-aritmética
»	877	Suma binaria
»	881	Resta binaria
»	881	Complementación
»	882	Multiplicación binaria
»	884	División binaria
»	885	Operaciones booleanas
»	888	Operaciones de desplazamiento
»	889	El registro de estado
»	890	Acarreo
»	890	Cero
»	890	Desbordamiento
»	891	Negativo
»	891	Medio acarreo
»	891	Paridad
»	892	Flags particulares
»	893	El buffer de la ALU
»	893	La unidad de control
»	896	Ejecución de las instrucciones
»	896	La temporización de las señales
»	903	La gestión de interrupción (interrupt)
»	905	La pila (stack)
»	910	Estructura de las instrucciones Assembler
»	911	Métodos de direccionamiento
»	912	Direccionamiento inmediato
»	912	Direccionamiento absoluto o directo
»	913	Direccionamiento indirecto
»	915	Direccionamiento indexado
»	915	Direccionamiento relativo
»	917	Direccionamiento de registro
»	917	Direccionamiento de pila
»	918	Las instrucciones Assembler
»	919	Instrucciones condicionales
»	920	Instrucciones aritméticas

Pág.	922	Instrucciones lógicas
»	922	Instrucciones de transferencia de datos
»	929	Instrucciones de bifurcación (branch)
»	930	Instrucciones de salto (skip)
»	931	Instrucciones de llamada a subrutina y retorno
»	932	Instrucciones varias
»	932	NOP
»	933	PUSH y POP
»	933	HALT y WAIT
»	933	Break
»	933	Enable Interrupt y Disable Interrupt
»	934	Adjust y Translate
»	934	Las reglas de ensamblaje
»	942	Ejemplo de programación en Assembler
»	945	El Assembler del Rockwell 6502
»	947	El Assembler del Zilog Z80
»	951	EL LENGUAJE COBOL
»	952	Generalidades
»	953	El programa fuente
»	955	Compilación y encadenado
»	957	La hoja de programación Cobol y las reglas de detalle
»	960	Estructura de un programa Cobol
»	964	IDENTIFICATION DIVISION
»	965	ENVIRONMENT DIVISION
»	965	CONFIGURATION SECTION
»	965	El párrafo SPECIAL-NAMES
»	966	El párrafo FILE-CONTROL
»	967	DATA DIVISION
»	967	FILE SECTION
»	968	Descripción de los datos: la cláusula BLOCK CONTAINS
»	968	La cláusula LABEL RECORD
»	969	La cláusula RECORDING MODE
»	970	La cláusula RECORD CONTAINS
»	970	Estructura de los records
»	973	WORKING-STORAGE SECTION
»	973	El nivel 77
»	974	El nivel 88
»	974	El nivel 66 y la cláusula RENAMES
»	975	La cláusula REDEFINES
»	976	La cláusula PICTURE
»	978	Descripción de los campos numéricos
»	979	Descripción de los campos alfabéticos
»	979	Descripción de los campos alfanuméricos
»	981	PROCEDURE DIVISION
»	988	Ejemplo de estructuración de una PROCEDURE DIVISION
»	992	Las instrucciones del Cobol
»	994	Verbos de I/O
»	995	Apertura y cierre de los files: los verbos OPEN y CLOSE
»	996	Utilización en lectura: OPEN INPUT
»	997	Utilización en escritura: OPEN OUTPUT
»	997	Utilización en lectura-escritura: OPEN I-O
»	997	Puesta en fila: OPEN EXTEND
»	1000	Lectura y escritura en files: los verbos READ y WRITE
»	1003	Escritura en files de impresión
»	1003	Cláusulas para el salto de líneas
»	1004	Cláusulas para el salto de página
»	1004	La cláusula LINAGE
»	1006	Las instrucciones ACCEPT y DISPLAY
»	1009	Verbos aritméticos
»	1009	Representación de los datos en memoria: la cláusula USAGE
»	1011	USAGE DISPLAY
»	1011	USAGE COMPUTATIONAL
»	1012	USAGE COMP-3

Pág.	1013	USAGE COMP-1 y COMP-2
»	1014	Las expresiones aritméticas
»	1015	La instrucción COMPUTE
»	1020	El verbo ADD
»	1022	El verbo SUBTRACT
»	1023	El verbo MULTIPLY
»	1023	El verbo DIVIDE
»	1025	Verbos de transferencia y de manipulación de los datos
»	1025	El verbo MOVE
»	1027	MOVE alfanumérica
»	1030	La cláusula ALL
»	1036	MOVE numérica
»	1040	La instrucción INSPECT
»	1040	La función de contado
»	1040	La cláusula ALL
»	1041	La cláusula LEADING
»	1041	La cláusula CHARACTERS
»	1041	La cláusula BEFORE
»	1042	La cláusula AFTER
»	1043	La función de sustitución
»	1044	Formato general de la instrucción INSPECT
»	1046	La instrucción STRING
»	1049	La instrucción UNSTRING
»	1051	Verbos de control
»	1052	La proposición IF
»	1053	Lógica de ejecución de la instrucción IF
»	1056	La opción NEXT SENTENCE
»	1057	Condiciones asociables a la instrucción IF
»	1058	Análisis de relación
»	1064	Análisis de signo
»	1064	Análisis de condición
»	1067	Análisis de clase
»	1068	Empleo de los operadores lógicos
»	1072	La instrucción GO TO
»	1075	La cláusula DEPENDING ON
»	1076	El verbo PERFORM
»	1084	La instrucción EXIT
»	1084	Ejecuciones iterativas del mismo procedimiento
»	1085	La cláusula UNTIL
»	1085	La cláusula VARYING... FROM... BY
»	1085	La instrucción STOP
»	1086	Gestión de las tablas en el Cobol
»	1089	Tablas de una dimensión
»	1095	Tablas de dos dimensiones
»	1100	Tablas de tres dimensiones
»	1103	Carga de una tabla
»	1107	Los índices de las tablas
»	1111	El verbo SET para la gestión de un índice
»	1112	El verbo SEARCH para la búsqueda en las tablas
»	1113	La instrucción SEARCH
»	1115	Uso de la SEARCH secuencial
»	1119	La instrucción SEARCH ALL (búsqueda binaria)
»	1121	Ordenado de una tabla
»	1124	Ordenado de los datos (Sort)
»	1127	La INPUT PROCEDURE y el verbo RELEASE
»	1127	La OUTPUT PROCEDURE y el verbo RETURN
»	1129	Gestión de los files de índices
»	1133	Descripción de los files IS en ENVIRONMENT DIVISION
»	1133	Descripción de los files IS en DATA DIVISION
»	1133	Uso de las instrucciones en la PROCEDURE DIVISION
»	1133	El verbo READ
»	1134	El verbo WRITE
»	1134	Instrucciones específicas para la gestión de los files IS
»	1135	El verbo REWRITE
»	1135	El verbo DELETE

Pág.	1136	El verbo START
»	1138	El verbo CLOSE
»	1138	Comunicación entre programas. Las subrutinas
»	1145	El verbo CALL
»	1145	PROCEDURE DIVISION de una subrutina
»	1145	La instrucción EXIT PROGRAM
»	1147	Ejemplos de aplicación
Volumen 5		
»	1153	EL LENGUAJE FORTRAN
»	1154	Características fundamentales en relación al Basic
»	1155	Operadores aritméticos, lógicos y relacionales
»	1156	VARIABLES Y CONSTANTES EN EL FORTRAN
»	1158	Declaraciones de tipo de las variables
»	1160	Dimensionado de las variables estructuradas
»	1160	Las equivalencias COMMON y EQUIVALENCE
»	1162	Instrucciones de asignación
»	1163	Las instrucciones del Fortran
»	1163	La instrucción GOTO
»	1164	Los bucles
»	1164	Bucles implícitos
»	1166	La forma DO... WHILE
»	1166	Instrucciones condicionadas
»	1168	Uso de las subrutinas
»	1169	Instrucciones de terminación y de paro de la ejecución
»	1170	Las funciones del Fortran
»	1170	Funciones de librería
»	1170	Conversión de tipo
»	1171	Funciones de cálculo
»	1172	Funciones de cadena
»	1173	Funciones matemáticas
»	1173	Funciones definidas por el usuario
»	1175	Las instrucciones y los formatos de I/O
»	1176	PRINT
»	1176	WRITE
»	1178	READ
»	1178	Formatos de I/O
»	1178	Formatos para los enteros
»	1179	Formatos para los reales
»	1180	Formatos para los caracteres
»	1180	Formatos para las constantes lógicas
»	1180	Descriptores de edit
»	1180	Edit en entrada
»	1180	Edit en salida
»	1180	Descriptores comunes
»	1180	Factor de escala
»	1181	Repeticiones de formato
»	1181	Instrucciones de I/O no formateadas
»	1181	Instrucciones de entrada
»	1182	Instrucciones de salida
»	1182	Ejemplo de programación en Fortran
»	1186	Gestión de los files
»	1190	Gestión de los files en disco
»	1191	OPEN
»	1193	READ y WRITE
»	1193	BACKSPACE
»	1193	REWIND
»	1193	ENFILE
»	1194	INQUIRE
»	1194	CLOSE
»	1194	Instrucciones particulares del Fortran
»	1194	EXTERNAL

Pág.	1196	ENTRY
»	1196	SAVE
»	1197	BLOCKDATA
»	1197	Implantaciones particulares
»	1198	Representación de las instrucciones en forma normal
»	1202	EL LENGUAJE PASCAL
»	1205	La programación estructurada
»	1205	La implantación de los programas
»	1206	El pseudocódigo
»	1207	Los diagramas de flujo estructurados
»	1212	Las estructuras de control
»	1213	Secuencia
»	1213	Selección
»	1214	Repetición
»	1216	Los datos en el Pascal
»	1217	Tipos de datos
»	1217	El tipo entero
»	1220	El tipo real
»	1221	El tipo carácter
»	1227	Tipo booleano
»	1228	Las declaraciones de tipo
»	1228	La declaración de contante
»	1228	La declaración de variable
»	1230	Tipo de datos escalares no estándar
»	1231	Tipos de datos definidos de nuevo y enumerados
»	1232	Tipos de datos subsistema
»	1233	Las primeras instrucciones del Pascal
»	1233	Expresiones aritméticas y booleanas
»	1234	Utilización de las funciones del Pascal
»	1235	La instrucción de asignación
»	1239	Las instrucciones de entrada: READ y READLINE
»	1240	Las instrucciones de salida: WRITE y WRITELN
»	1243	Estructura de un programa Pascal
»	1249	Las instrucciones de control
»	1249	Las instrucciones de iteración
»	1249	La instrucción FOR... TO... DO...
»	1251	La instrucción WHILE... DO...
»	1252	La instrucción REPEAT... UNTIL...
»	1254	Las instrucciones condicionales
»	1254	La instrucción IF... THEN... ELSE...
»	1257	La instrucción CASE... OF...
»	1259	La instrucción de salto condicionado: GOTO...
»	1260	Aspectos particulares del Pascal
»	1262	Ejemplos de programación en Pascal
»	1266	RECAPITULACIÓN: DESARROLLO DE UN PROCESO DE FACTURACION
»	1266	Análisis del problema
»	1273	Files utilizados
»	1273	Menú principal
»	1277	Gestión de los archivos
»	1286	Emisión de las facturas
»	1303	Impresión del sellado
»	1305	Otras funciones
»	1316	LA TRANSMISION DE LAS INFORMACIONES
»	1317	Modalidades y sistemas de comunicación
»	1320	Canales simplex, half-duplex, full duplex
»	1321	Comunicaciones por línea telefónica
»	1321	Características de las señales
»	1324	Técnicas de modulación. Los modems
»	1326	Configuraciones de las conexiones

Pág.	1328	Configuración punto a punto
»	1328	Configuración multipunto
»	1329	Configuraciones de multiplexador
»	1330	Interfaz de comunicación
»	1335	Comunicaciones a cortas distancias
»	1337	Protocolos de comunicación
»	1339	El handshake
»	1340	El método XON/XOFF
»	1341	El método ENQ/ACK
»	1349	El protocolo BSC
»	1351	El protocolo HDLC
»	1353	El protocolo IEEE-488
»	1355	Redes para comunicación de datos
»	1356	Topología y características de las redes
»	1356	Red en cadena
»	1356	Red en estrella
»	1356	Red en anillo
»	1356	Red de árbol o jerárquica
»	1360	Redes públicas
»	1360	Redes locales
»	1362	Arquitecturas que surgen
»	1364	Instrucciones del Basic para la comunicación de datos
»	1368	SISTEMAS OPERATIVOS
»	1368	Monousuario y multiusuario
»	1369	Batch e interactividad
»	1369	Multiprogramación
»	1371	Tiempo compartido
»	1372	Tiempo real
»	1373	Sistema file
»	1374	Estructura sistema file del UNIX
»	1378	Procesos concurrentes
»	1380	Comunicaciones entre procesos
»	1383	LOS GRAFICOS DE ORDENADOR EN BASIC
»	1384	Aplicaciones del proceso gráfico
»	1384	Gráficos de gestión
»	1384	Proyecto
»	1384	El proyecto mecánico
»	1385	El proyecto electrónico
»	1385	El proyecto arquitectónico
»	1386	Diseño industrial
»	1386	Proceso de imágenes (image processing)
»	1387	Animación
»	1387	Periféricos orientados a los gráficos
»	1388	Dispositivos de entrada
»	1389	La mesa gráfica
»	1394	Dispositivos especiales
»	1396	Dispositivos de salida
»	1396	La impresora gráfica
»	1398	El plotter
»	1404	El monitor monocromático
»	1408	El monitor en colores
»	1411	Hardware especializado
»	1414	Instrucciones Basic para los gráficos
»	1414	Trazado de segmentos
»	1418	Rotación de un segmento
»	1427	Trazado de una circunferencia
»	1432	Presentación del gráfico de una función
»	1433	Trazado del gráfico de una función por segmentos
»	1436	Trazado de una recta

Volumen 6

Pág.	1441	Presentación de los ejes cartesianos
»	1443	Trazado de la recta que pasa por dos puntos
»	1448	Regresión lineal de mínimos cuadrados
»	1462	Programa para el trazado del gráfico de una función cualquiera
»	1465	Cálculo y normalización del factor de escala
»	1465	Búsqueda de los valores máximo y mínimo de Y
»	1465	Presentación del proceso
»	1465	Gestión de los errores
»	1482	Presentación de caracteres en el modo gráfico
»	1502	Histogramas y diagramas de tarta
»	1530	Empleo de la memoria en el funcionamiento en modalidad gráfica
»	1530	La memoria vídeo
»	1532	Direccionado de la memoria vídeo
»	1533	Direccionado según el eje Y
»	1537	Direccionado según el eje X
»	1539	Aplicaciones del direccionado directo de la memoria vídeo
»	1543	Memorización de imágenes gráficas
»	1555	Cálculo de áreas y perímetros
»	1564	Instrucciones Basic para la memorización de figuras
»	1564	POINT
»	1564	GET
»	1565	PUT
»	1565	Las ventanas vídeo
»	1566	Programa para la generación de ventanas vídeo
»	1587	Instrucciones Basic para la gestión de las ventanas vídeo
»	1590	Vectores gráficos y tablas de las figuras
»	1590	Codificación de los vectores de desplazamiento
»	1597	Programa para la generación y la gestión de tablas de las figuras
»	1605	Gestión de los desplazamientos de conjunto (subrutina 4100)
»	1605	Desplazamiento de la figura (subrutina 6000)
»	1610	Rotación de la figura (subrutina 4500)
»	1611	Transferencia al disco (subrutina 5000)
»	1611	Lectura del disco (subrutina 5200)
»	1614	Programas de utilidad para la creación de figuras gráficas
»	1614	Shape Editor
»	1623	Shape Loader
»	1623	Instrucciones Basic para la gestión de figuras gráficas
»	1623	DRAW
»	1631	XDRAW
»	1632	ROT
»	1632	SCALE
»	1633	Gráficos tridimensionales
»	1634	Las funciones de dos variables
»	1635	Representación tridimensional
»	1637	Los problemas gráficos en tres dimensiones
»	1658	La animación de imágenes en el ordenador
»	1658	Los sprites
»	1658	Gestión software de los sprites
»	1658	La animación de imágenes gráficas
»	1661	Gestión del fondo
»	1661	Presentación de un sprite
»	1663	Problemas correspondientes a la animación
»	1669	Gestión hardware de los sprites
»	1669	La memorización del sprite
»	1673	La presentación del sprite
»	1674	El desplazamiento del sprite
»	1674	Empleo de la memoria para la gestión de los sprites
»	1674	Ejemplo de aplicación
»	1679	Gestión avanzada de los sprites
»	1679	Definición de los sprites
»	1679	Presentación
»	1685	Control de las colisiones
»	1688	Glosario

Indice de las palabras reservadas

FUNCIONES DEL SISTEMA OPERATIVO

SEARCH = p. 304
OPEN = p. 304
CLOSE = p. 304
RENAME = p. 304
READ = p. 305
WRITE = p. 305
SELECT = p. 305
ERA = p. 305
DIR = p. 305
REN = p. 307
SAVE = p. 307
TYPE = p. 307
STAT = p. 307
ASM = p. 307
DDT = p. 307
LOAD = p. 307
PIP = p. 310
ED = p. 310
FORMAT = p. 310
SYSTEM = p. 311

BASIC

NOT, AND, OR, XOR (operadores lógicos) = p. 326, 327, 331
DEF FNA (operador funcional) = p. 332 a 335
>MBASIC = p. 354
NEW = p. 354
AUTO = p. 353, 354
SAVE = p. 353, 354
EDIT = p. 356, 357, 358
CSAVE, CLOAD = p. 363
LIST, LLIST = p. 363
CONT = p. 363
TRON, TROFF = p. 363
MERGE = p. 364
CLEAR = p. 364
OPTION BASE 1 = p. 368
REM = p. 368
DEFINT, DEFSTR, DEFDBL, DEFSTR = p. 370
LET = p. 370
PRINT = p. 371
INPUT = p. 371, 548 a 550
DATA, READ, RESTORE = p. 372, 375, 379 a 382
LSET, RSET = p. 382
SPACES\$(N) = p. 382
SPC(N) = p. 384
SWAP = p. 385
FOR... NEXT = p. 385
WHILE... WEND = p. 393
GOTO... = p. 397
ON... GOTO... = p. 397
IF... THEN... ELSE = p. 408
GOSUB... = p. 411
CHAIN... = p. 411
ALL = p. 412
COMMON = p. 412
DELETE = p. 412
RETURN = p. 417
ON ERROR GOTO... = p. 418
ABS(R) = p. 441
CDBL(R) = p. 441
CINT(R) = p. 442
CSNG(R) = p. 442

FIX(R) = p. 442
INT(R) = p. 443
SNG(R) = p. 443
SQR(R) = p. 443
RND(R) = p. 444
RANDOMIZE = p. 446
ATN(Y) = p. 447
COS(R) = p. 447
SIN(R) = p. 447
TAN(R) = p. 447
EXP(R) = p. 449
LOG(R) = p. 450
ASC(A\$) = p. 450
CHR\$(N) = p. 450
CVI(A\$) = p. 452
CVS(A\$) = p. 454
CVD(A\$) = p. 454
HEX\$(N) = p. 454
INKEY\$ = p. 454
INPUT\$(N) = p. 455, 554
INSTR(N,A\$,B\$) = p. 455
LEFT\$(A\$,N) = p. 457
LEN(A\$) = p. 458
MID\$(A\$,NS,NC) = p. 458
MKI\$(N) = p. 460
MKM\$(R) = p. 460
MKD\$(D) = p. 460
OCT\$(N) = p. 460
RIGHT\$(A\$,N) = p. 460
STR\$(R) = p. 465
STRING\$(N,K) = p. 468
VAL(A\$) = p. 469
EOF(N) = p. 469
FRE(A) = p. 470
INP(N) = p. 470
LOC(N) = p. 470, 646
LPOS(N) = p. 471
OUT N,M = p. 472
PEEK(N) = p. 472
POKE N,M = p. 472
POS(N) = p. 473
DIM NOMBRE (N) = p. 520 a 522
LINE INPUT = p. 555
LPRINT = p. 556
TAB(N) = p. 557
LPRINT USING = p. 557
OPEN = p. 637, 642
INPUT # = p. 638
LINE INPUT # = p. 638
WRITE # = p. 638
PRINT # y PRINT USING # = p. 639
EOF(N) = p. 639
LOC(N) = p. 639
KILL = p. 639
NAME = p. 639
FIELD = p. 643
CLOSE # = p. 643
PUT # = p. 643
GET # = p. 646

COMPILACION

BASCOM.COM = p. 717
BRUN.COM = p. 717
BASLIB.REL = p. 719
OBSLIB.REL = p. 719
BCLOAD = p. 719
L80.COM = p. 719

INSTRUCCIONES PARTICULARES DEL BASIC 80

CALL = p. 733
COMMON = p. 736
USR = p. 736
STOP = p. 738
STICK (n) = p. 741
STRING = p. 741
ON STRING... = p. 741
PEN ON, OFF = p. 741
Z = PEN (N) = p. 741
Compendio del Basic 80 = p. 742 a 745
ALLOCATE = p. 746
DEALLOCATE = p. 746

COMANDO DOS

CATALOG = p. 747
LOAD = p. 747
SAVE = p. 747
INIT = p. 747
DELETE = p. 748
LOCK = p. 748
RENAME = p. 748
VERIFY = p. 748
Comandos tablero electrónico = p. 800, 801
Funciones tablero electrónico = p. 814 a 823
Resumen funciones Multiplan y Visicalc = p. 845, 846

EL LENGUAJE ASSEMBLER

ADD A, # = p. 912
ADD A, / = p. 912
ADDC = p. 919
BGT = p. 919
Caracteres de especificación = p. 919
Z = p. 919
NZ = p. 919
Tests condicionales = p. 920
Instrucciones aritméticas = p. 921
Instrucciones lógicas = p. 923, 924
Instrucciones de transferencia de datos = p. 926, 927
Instrucciones de salto = p. 928, 929
Instrucciones de llamada a subrutinas = p. 932
Instrucciones de retorno = p. 932
Instrucciones varias = p. 933
Reglas de ensamblaje = p. 934
Instrucciones Assembler del microprocesador Rockwell 6502 = p. 947
Instrucciones Assembler del microprocesador Zilog Z80 = p. 949, 950

EL LENGUAJE COBOL

Constantes figurativas = p. 993
Verbos OPEN y CLOSE = p. 995

Verbos READ y WRITE = p. 1000
Instrucciones ACCEPT y DISPLAY = p. 1006
USAGE = p. 1009
USAGE DISPLAY = p. 1011
USAGE COMPUTATIONAL = p. 1011
USAGE COMP-3, COMP-1, COMP-2 = p. 1012, 1013
Expresiones aritméticas = p. 1014
COMPUTE = p. 1015
Verbo ADD = p. 1020
Verbo SUBTRACT = p. 1022
Verbo MULTIPLY = p. 1023
Verbo DIVIDE = p. 1023
Verbo MOVE = p. 1025
Instrucción MOVE = p. 1039
Instrucción INSPECT = p. 1040
Instrucción STRING = p. 1046
Instrucción UNSTRING = p. 1049
Proposición IF = p. 1052
Instrucción GO TO = p. 1072
Verbo PERFORM = p. 1076
Verbo SEARCH = p. 1112
INPUT PROCEDURE y verbo RELEASE = p. 1127
OUTPUT PROCEDURE y verbo RETURN = p. 1127
Instrucciones en la PROCEDURE DIVISION = p. 1133
Instrucciones específicas para la gestión de los files IS = p. 1134
Operaciones realizables en files IS = p. 1138
Verbo CALL = p. 1145
Instrucción EXIT PROGRAM = p. 1145

EL LENGUAJE FORTRAN

TRUE y FALSE = p. 1157
Operadores lógicos y relacionales = p. 1157
Variables (declaraciones de tipo) = p. 1158
INTEGER = p. 1158
REAL = p. 1158
DOUBLE PRECISION = p. 1158
COMPLEX = p. 1158
LOGICAL = p. 1158
CHARACTER = p. 1158
Variables resultantes de los cálculos = p. 1159
IMPLICIT, IMPLICIT CHARACTER = p. 1160
DIMENSION = p. 1160
COMMON = p. 1160
EQUIVALENCE = p. 1160
DATA = p. 1162, 1163
PARAMETER = p. 1163
La instrucción GOTO = p. 1163, 1164
DO... WHILE... = p. 1166
IF = p. 1166, 1167
CALL = p. 1168
PAUSE, STOP, END = p. 1169
Conversiones de tipo de variables = p. 1170 a 1172
FUNCTION = p. 1173
PRINT = p. 1176
WRITE = p. 1176, 1177
READ = p. 1178
Formatos I/O = p. 1178
OPEN = p. 1191
READ y WRITE = p. 1193

BACKSPACE = p. 1193
REWIND = p. 1193
ENDFILE = p. 1193
INQUIRE = p. 1194
CLOSE = p. 1194
EXTERNAL = p. 1194
ENTRY = p. 1196
SAVE = p. 1196
BLOCKDATA = p. 1197

EL LENGUAJE PASCAL

Funciones = p. 1221
CHR (), ORD (), PRE (), SUCC () = p. 1221
Declaraciones de tipo = p. 1228
Expresiones aritméticas y booleanas = p. 1233, 1234
Instrucciones de asignación = p. 1235, 1236
READ y READLN = p. 1239
WRITE y WRITELN = p. 1240
FOR... TO... DO... = p. 1249
WHILE... DO... = p. 1251
REPEAT... UNTIL... = p. 1252
IF... THEN... ELSE = p. 1254
CASE... OF... = p. 1257

INSTRUCCIONES BASIC PARA LA COMUNICACION DE DATOS

TIME \$ = p. 1367

Indice de los listados y programas de ejemplo

Subrutinas para el cálculo de áreas y perímetros = p. 373
Ejemplos sobre el uso de RESTORE = p. 380
Ejemplos sobre el uso de READ y DATA = p. 381
Ejemplos sobre el uso de las instrucciones SPACE\$, LSET, RSET = p. 383
Lectura de los datos e impresión en forma tabular = p. 386
Ejemplo de bucle con valores no enteros = p. 388
Ejemplo de un bucle interpretado y compilado = p. 390
Ejemplo de tres bucles nidificados = p. 391
Subrutina de generación de cadenas = p. 392
Ejemplo de utilización de la rutina 1000 = p. 392
Programa para el uso de la instrucción WHILE... WEND = p. 395
Cálculo de conveniencia entre diesel y gasolina = p. 415
Programa de agenda = p. 429
Programa para el cómputo del interés = p. 442
Programa para el redondeo de los números = p. 444
Programa para la generación de números aleatorios = p. 444
Ejemplo de uso de las funciones trigonométricas = p. 447
Cálculo de la distancia de un barco a la costa = p. 449

INSTRUCCIONES BASIC PARA LOS GRAFICOS

HPLOT = p. 1414
LINE = p. 1414
LINE... STEP = p. 1416
COLOR = p. 1423
CIRCLE = p. 1427
TEXT = p. 1518
HOME = p. 1522
VTAB n = p. 1522
HTAB m = p. 1522
HGR2 = p. 1522
HCOLOR = n = p. 1522
HPLOT X1, X1 TO X2, Y2 = p. 1522
GET A\$ = p. 1522
PEEK = p. 1538, 1543
HIMEM y LOMEM = p. 1546, 1547
POINT = p. 1564
GET = p. 1564
PUT = p. 1565
HGR = p. 1565
WINDOW = p. 1587 a 1589

Ejemplo de uso de la sentencia ASC (A\$) = p. 450
Programa para el uso de la sentencia CHR\$ = p. 452
Uso de la función HEX\$(N) = p. 455
Cálculo de las "recurrencias" de un carácter en una cadena = p. 457
Programa para la selección de datos de un archivo = p. 460
Ejemplo de uso de la sentencia MID\$ = p. 461
Programa de conversión = p. 462
Programa de aplicación para el uso de la subrutina 1000 = p. 464
Programa de aplicación para el uso de la subrutina 2000 = p. 466, 467
Programa que utiliza una subrutina = p. 476
Programa que utiliza una función definida por el usuario = p. 476
Programa para el uso de una función de cadena = p. 482
Solución de ecuaciones de segundo grado = p. 484
Programa de control y conversión de caracteres = p. 487
Programa para el cálculo de áreas elementales = p. 491
Programa de cálculo de las disipaciones térmicas = p. 505
Asignación por matriz bidimensional = p. 526

Asignación por matriz tridimensional = p. 526
Programa de simulación de errores = p. 529
Ejemplo de cálculo utilizando cadenas de matriz = p. 534
Programa para el cálculo de medias = p. 539
Programa para el cálculo de la desviación cuadrática media = p. 544
Ejemplo de uso de la función INKEY\$ = p. 552
Ejemplos de uso de la función INPUT\$(N) = p. 555
Ejemplos de uso de la instrucción LPRINT = p. 556
Ejemplos de uso de la instrucción LPRINT USING = p. 559
Programas para tabulados de 80 columnas = p. 564 a 566
Programa de impresión parametrizada = p. 574
Ejemplos de gestión de la impresora = p. 583
Programa de impresión de diagramas de barras = p. 592
Ejemplos de gestión del cursor = p. 597
Programa de introducción y control de fechas = p. 604
Ejemplo de activación de las teclas funcionales = p. 608
Menú principal = p. 625
Menú del almacén = p. 628
Ejemplo de preparación de histogramas = p. 634
Ejemplo de lectura y escritura de un file secuencial = p. 641
Programa que escribe en un file secuencial = p. 647
Programa que escribe en un file directo = p. 655
Preparación, escritura y lectura de un record = p. 657
Creación del file e inicialización del último record = p. 668
Subrutina de apertura del file = p. 669
Subrutina de introducción de datos = p. 669
Subrutina de actualización = p. 670
Ordenamiento de una matriz = p. 678, 679
Preparación de descripciones y longitudes de campos = p. 698
Presentación de las máscaras vídeo = p. 702
Gestión máscaras de vídeo = p. 706
Gestión disco parametrizada = p. 712
Main de prueba = p. 715
Ejemplo de compilación = p. 721
Ejemplo de uso de POKE y PEEK = p. 739
Ejemplo de gestión de los files secuenciales (DOS) = p. 748
Operaciones de I/O sobre files secuenciales con valores numéricos = p. 750
Ampliación y relectura de un file secuencial = p. 751
Ejemplo de uso de la instrucción ONERR (DOS) = p. 752
Uso de la instrucción GET = p. 753
Programa de ejemplo de acceso a files directos = p. 754
Programa para el control de una contraseña = p. 792
Estructuración de una PROCEDURE DIVISION (1) = p. 990
Estructuración de una PROCEDURE DIVISION (2) = p. 991
Ejemplo de apertura de un file = p. 996
Ejemplo de uso de los verbos OPEN y CLOSE = p. 999
Ejemplo de uso de las instrucciones ACCEPT y DISPLAY = p. 1007
Ejemplo de aplicación del verbo COMPUTE = p. 1018
Ejemplo de uso de la cláusula BEFORE = p. 1042
Conversión de una fecha = p. 1047
Aplicación de la instrucción IF = p. 1055
Lectura y proceso de un file fichas = p. 1061
Cálculo de las raíces de una ecuación de segundo grado = p. 1066
Ejemplo de uso del operador AND = p. 1068
Ejemplo de uso de los operadores AND y OR = p. 1072
Ciclo de lectura de un file secuencial = p. 1075
Lectura y proceso de un file (1) = p. 1088
Lectura y proceso de un file (2) = p. 1090
Lectura y proceso de un file (3) = p. 1093
Empleo de una tabla de dos dimensiones = p. 1099
Ejemplo de impresión de los datos de una tabla de dos dimensiones = p. 1102
Programa que utiliza la carga y la gestión de una tabla = p. 1108
Descripción de una tabla con índice gestionado por el compilador = p. 1111
Búsqueda en tabla mediante uso de subindexado = p. 1114
Búsqueda en tabla mediante SEARCH secuencial = p. 1115
Búsqueda secuencial en una tabla = p. 1118
Búsqueda dicotómica en una tabla = p. 1122

Ejemplo de ordenado y selección de datos = p. 1126
Ejemplo de uso de la instrucción RELEASE = p. 1128
Ejemplo de uso de la instrucción RETURN = p. 1130
Estructura de un programa llamado = p. 1146
Estructura de un programa llamado = p. 1146
Ejemplo de aplicación: control de una fecha = p. 1148
Ejemplo de aplicación: conversión de una fecha = p. 1151
Programa de comparación Basic-Fortran = p. 1184
Subrutina de introducción de datos (comparación Basic-Fortran) = p. 1186
Subrutinas de control y cálculo (comparación Basic-Fortran) = p. 1189
Subrutina de impresión = p. 1191
Lectura de un file y cálculo de la media de los valores = p. 1196
Uso de las funciones PRED y SUCC = p. 1263
Aplicaciones de las instrucciones READ y READLN = p. 1264
Aplicaciones de la instrucción WHILE... DO... = p. 1265
Desarrollo de un proceso de facturación = p. 1266
Procedimiento de facturación. Gestión archivos = p. 1287
Programa de ordenado (main) = p. 1308
Programa de ordenado (subrutinas) = p. 1313
Trazado de un segmento = p. 1416
Trazado parametrizado de cuadrados y rectángulos = p. 1421
Rotación de un segmento = p. 1426
Trazado de una circunferencia = p. 1430
Trazado de una recta = p. 1438
Trazado de una recta que pasa por dos puntos = p. 1446
Trazado de la recta de regresión = p. 1456
Trazado del gráfico de una función = p. 1473
Presentación de cifras en modo gráfico = p. 1489
Presentación de caracteres en modo gráfico = p. 1496
Presentación de histogramas = p. 1504
Presentación de diagramas de tarta = p. 1527
Direccionado memoria gráfica = p. 1542
Lectura y presentación de la memoria vídeo = p. 1546
Memorización de imágenes gráficas en disco = p. 1553
Programa para el cálculo de áreas = p. 1561
Creación y gestión de las ventanas vídeo = p. 1579
Gestión de una tabla de las figuras = p. 1596
Creación y gestión de tablas de las figuras = p. 1612
Shape Editor = p. 1618
Shape Loader = p. 1626
Gráfico de una función de dos variables = p. 1640
Gráfico tridimensional con borrado de las líneas ocultas = p. 1653
Ejemplo de gestión de los sprites por software = p. 1667
Ejemplo de uso de los sprites CBM 64 = p. 1681
Gestión de los sprites bajo MSX = p. 1685

Indice de los tests

Número 1 = p. 57 (temas: calculadores, números binarios, decimales, octales, hexadecimales)
 Soluciones número 1 = p. 76
 Número 2 = p. 106 (temas: tablas de verdad)
 Soluciones 2 = p. 122
 Número 3 = p. 124 (temas: código ASCII, métodos de transmisión, impresora)
 Soluciones 3 = p. 150
 Número 4 = p. 152 (temas: contador de programa, CPU, S.O., DMA, mapa de memoria, Interrupt, memorias)
 Soluciones 4 = p. 183
 Número 5 = p. 206 (temas: simbología de diagramas de flujo)
 Soluciones 5 = p. 214
 Número 6 = p. 240 (temas: memorias de masa)
 Soluciones 6 = p. 244
 Número 7 = p. 279 (temas: files, Sort, búsqueda serie y dicotómica)
 Soluciones 7 = p. 286
 Número 8 = p. 311 (temas: componentes principales de un S.O., tablas de los extend, directorio, formateado)
 Soluciones 8 = p. 313
 Número 9 = p. 335 (temas: modo inmediato, campos)
 Soluciones 9 = p. 350
 Número 10 = p. 365 (temas: comando SAVE, líneas de programa)
 Soluciones 10 = p. 374
 Número 11 = p. 385 (temas: constantes, matrices, MOD)

Soluciones 11 = p. 398
 Número 12 = p. 410 (temas: definición tipo variable, DATA)
 Soluciones 12 = p. 413
 Número 13 = p. 446 (temas: bloque de programa, variables)
 Soluciones 13 = p. 451
 Número 14 = p. 468 (temas: cadenas)
 Solución 14 = p. 473
 Número 15 = p. 535 (temas: funciones, cálculos combinatorios)
 Soluciones 15 = p. 545
 Número 16 = p. 560 (temas: matrices)
 Soluciones 16 = p. 567
 Número 17 = p. 593 (temas: impresoras)
 Soluciones 17 = p. 599
 Número 18 = p. 637 (temas: Basic, funciones)
 Soluciones 18 = p. 644
 Número 19 = p. 663 (temas: files de datos)
 Soluciones 19 = p. 671
 Número 20 = p. 733 (temas: records, Compiladores)
 Soluciones 20 = p. 737
 Número 21 = p. 1237 (temas: Pascal)
 Soluciones 21 = p. 1248
 Número 22 = p. 1338 (temas: transmisión de datos)
 Solución 22 = p. 1342
 Número 23 = p. 1363 (temas: protocolos transmisiones)
 Soluciones 23 = p. 1367

Indice de los temas complementarios

¿Puede pensar un ordenador? = p. 18
 El tejedor de números: historia de un descubrimiento = p. 24
 El ordenador en la familia = p. 40
 El ordenador en el colegio = p. 48
 La revolución informática = p. 58
 Un terminal en el televisor = p. 68
 Cómo se pregunta a un ordenador = p. 84
 El banco del futuro = p. 92
 Códigos secretos contra los piratas del software = p. 102
 Memorias que no olvidan = p. 134
 Funcionamiento de las memorias sólo en lectura = p. 144
 Las máquinas que leen = p. 164
 La revolución de la imprenta = p. 180
 Los circuitos integrados = p. 194
 Las máquinas que hablan = p. 216
 El hombre que inventó los videojuegos = p. 234
 La oficina computerizada = p. 252
 Bancos de datos especializados = p. 280
 La animación con ordenador = p. 300
 Modelado por ordenador = p. 328
 Wafer, chip & Co. = p. 359
 Andante con bit para ordenador solo = p. 376
 ¿Cuál es el futuro del ordenador? = p. 402
 Ha nacido la turismática = p. 420
 El hombre frente a la máquina = p. 431

Un ordenador tras las ruedas = p. 477
 ...Y como maestro, la Tortuga = p. 492
 La vaca lechera computerizada = p. 516
 La inteligencia artificial = p. 540
 Ordenador y psique = p. 584
 El problema de la reserva = p. 611
 Algunos kbytes para jugar = p. 648
 Un ordenador personal para comunicarse = p. 683
 Sammie y el automóvil = p. 727
 Del silicio al ordenador = p. 763
 ¿Qué software? = p. 777
 La automatización del trabajo de oficina (1) = p. 807
 La automatización del trabajo de oficina (2) = p. 827
 La automatización del trabajo de oficina (3) = p. 852
 El diagnóstico computerizado (1) = p. 935
 El diagnóstico computerizado (2) = p. 983
 El diagnóstico computerizado (3) = p. 1031
 El diálogo del hombre con las máquinas = p. 1078
 Las previsiones del ordenador = p. 1140
 Cómo se proyecta un videojuego inteligente (1) = p. 1222
 Cómo se proyecta un videojuego inteligente (2) = p. 1269
 Juegos de guerra para profesionales = p. 1343
 El ordenador y los hombres radar = p. 1511
 La anatomía del ordenador personal = p. 1548
 El ordenador y la vela = p. 1583

Corrección de erratas: las letras A y B indican, respectivamente, la columna de la izquierda y de la derecha. El número después de la coma indica la línea. Se han empleado las siguientes abreviaturas: t. = tabla; g. = gráfico; l. = listado.

ERRATA	CORRECCION	
27, B, 27	en 1885	en 1865
27, B, 33	Babbage falleció en 1871, a los ochenta y tres	Babbage falleció en 1871, a los setenta y nueve
45, B, g. 5	TB ← T → TB	TS ← T → TB
52, B, 21	pueden valer de 1 a 15	pueden valer de 0 a 15
77, test, 29	3 = 111	3 = 11
77, test, 32	3FB = 11111111011 = 1019 decimal	3FB = 1111111011 = 1019 decimal
81, B, 22	Operador OR	Operador AND
89, g. 3	Entrada B OR exclusivo no inversor	Entrada B OR no inversor
89, g. 6	Entrada B OR exclusivo inversor	Entrada B OR inversor
104, l./g., 5	CLAVE DE TRABAJO C F Z A L H T R P A...	CLAVE DE TRABAJO C F Z A L H C K P A...
105, B, l./g., 7	Transmisión codificada Forma de onda	11010 10001 00011 11011 10100 11000 10111 01000 01100
106, test, 2	A B A XOR B A AND (A XOR B)	A B A XOR B A AND (A XOR B)
112, B, 23, 24	las distintas listas tengan entre ellas un margen de página en blanco.	las distintas listas tengan entre ellas una página en blanco.
113, l./g., 28	26 1A SU	26 1A SUB
114, A, l./g., 9	39 27'	39 27/
114, A, l./g., 14	44 2C'	44 2C'
114, A, l./g., 34	64 40 C	64 40 @
114, B, l./g., 28	92 5C	92 5C \
114, B, l./g., 31	95 5F <	95 5F -
114, B, l./g., 32	96 60	96 60 \
115, B, l./g.	QUESTO E \ UN ESEMPIO 51 55 45 53 54 4F 20 45 2C 20 55 4E 20 45 53 45 4D 50 49 4F	ESTO ES UN EJEMPLO 45 53 4F 20 45 53 20 55 4E 20 45 4A 45 4D 50 4C 4F
116, l./g.		

TABLA CODIGOS ASCII

7 6 5 4 3 2 1	COLUMNA FILAS	000	001	010	011	100	101	110	111
		0	1	2	3	4	5	6	7
0 0 0 0	0	NUL	DLE	SP	0	@	P	.	P
0 0 0 1	1	SOH	DC 1	!	1	A	Q	a	q
0 0 1 0	2	STX	DC 2	"	2	B	R	b	r
0 0 1 1	3	ETX	DC 3	#	3	C	S	c	s
0 1 0 0	4	EOT	DC 4	\$	4	D	T	d	t
0 1 0 1	5	ENO	NAK	%	5	E	U	e	u
0 1 1 0	6	ACK	SYN	&	6	F	V	f	v
0 1 1 1	7	BEL	ETB	'	7	G	W	g	w
1 0 0 0	8	BS	CAN	(8	H	X	h	x
1 0 0 1	9	HT	EM)	9	I	Y	i	y
1 0 1 0	10 (A)	LF	SUB	+	:	J	Z	j	z
1 0 1 1	11 (B)	VT	ESC	+	;	K	[k	{
1 1 0 0	12 (C)	FF	FS	.	<	L	\	l	
1 1 0 1	13 (D)	CR	GS	-	=	M]	m	}
1 1 1 0	14 (E)	SO	RS	.	>	N	^	n	~
1 1 1 1	15 (F)	SI	US	/	?	O	_	o	DEL

121, A, 28	= (mensaje recibido) AND 1	= (mensaje recibido) AND 00000001
121, A, 30	= (mensaje recibido) AND (11111110)	= (mensaje recibido) AND 11111110
121, A, 32,33	tre un número binario cualquiera (dato recibido) y el valor 1, el resultado es el primer bit del dato	tre un número binario cualquiera (dato recibido) y el valor 1, el resultado del último bit del dato
121, A, 35	Por ejemplo, en el caso de la letra F = 1000110:	Por ejemplo, en el caso del mensaje 1000110:
121, A, 43	Dato AND 1:	Mensaje AND 00000001:
121, B, 1,2	Sin embargo, en el caso de la letra H = 1001000:	Sin embargo, en el caso del mensaje 1001000:
121, B, 10	Dato AND 1:	Mensaje AND 00000001:
122, Sol. test, 9	Obsérvese que a AND (A XOR B) es...	Obsérvese que A AND (A XOR B) es...
122, Sol. test, 26	NOT, de la pág. 80 a pág. 83 y de pág. 88 a pág. 90.	NOT, de la pág. 80 a la pág. 83 y y de la pág. 88 a la pág. 91.
123, t./g., 8	0 1 1	0 0 1
123, t./g., 16	A F A NAND F B B OR (A NAND F)	A G A NAND B B OR (A NAND F)
124, t./g., 3	3/Escribir... carácter K.	3/Escribir... carácter K (transmisión serie asíncrona).
124, t./g., 4	4/Traducir en códigos binarios las...	4/Traducir en códigos decimales las...
124, t./g., 8	d) escritura de los mismos... e) salto de página	d) escritura de los mismos... e) salto de página
124, t./g., 11, 12, 13	6/Escribir... los bits, 3, 7 y 1... numerados... Comprobar... con los números decimales 12, 21, 6 y 5.	6/Escribir... los bits 7, 3 y 1... numerados... Comprobar la exactitud de la solución aplicando la máscara sobre la representación binaria de los números decimales 12, 21, 6 y 15.
150, t./g., 22	binario 1001011... de paridad es 0.	binario 1001011... de paridad es 1.
150, t./g., 24	(MARK), luego...	(MARK = 1), luego...
150, t./g., 25	0 1 0 0 1 0 1 1 0 1 1	0 1 0 0 1 0 1 1 1 1 1
151, t./g., 10	z (= 122) = Z (90) + 32	122 (z) = 90 (Z) + 32
152, t./g., 4	instrucción a seguir.	instrucción a ejecutar.
152, t./g., 7	ejecuta después de la 103...	ejecuta después de la 106...
153, t./g., 15	b) La unidad... alrededor de 64.000 caracteres.	b) La unidad... alrededor de 32.000 caracteres.
158, B, 14	las existencias actuales)	las existencias actuales y nuevo costo unitario)
161, t./g., 31	A XY 35 270 8450	A XY 35 270 9450
161, t./g., 34	Total General 390450	Total General 391450
162, t./g., 26 a 29	Impresión de existencias y mano de obra para cada artículo	Impresión de existencias y valor para cada artículo
178, t./g., 19	CODICE	CODIGO
250, 19	151).	251).
271, A, 28	en la pág. 263	en la pág. 262
271, A, 46, 47	Traduciendo por	Disponiendo en
323, B, 7	da en pantalla el resultado 4.	da en pantalla el resultado 5.
347, t./g., 8	Los valores son: N(0) = 30 N(1) = 11 N(2) = 5	Los valores son: N(0) = 5 N(1) = 11 N(2) = 30
369, A, t./g., 19	50 y 95	60 y 95
388, t./g., 7	160' Paso = 0.1 se convierte en 10	160' Paso = 0.1 se convierte en 1

419, t./g., 14	Subrutinas de corrección de errores (error handling subrutinas)	Subrutinas de corrección de errores (error handling subrutinas)
443, A, 13	se convierte en -373	se convierte en -374
471, t./g., 17	Pri	Prime
501, B, 12	1.5 para el este;...	1.15 para el este;...
529, t., 15	130 FOR I=1 TO 4 'Lectura de la cadena D\$	130 FOR I=1 TO 4
529, t., 16	140 READ D\$ (I)	140 READ D\$ (I) 'Lectura de la cadena D\$
529, t., 28	250 FOR I = 1 TO 3 'Lectura de las cadenas R\$	250 READ D\$ (I)
529, t., 29	260 READ R\$ (I)	260 READ R\$ (I) 'Lectura de las cadenas R\$
529, t., 39	360... en las filas 120 y 140 no son	360... en las filas 120 y 240 no son
569, t./g., 21	cuál de los 5 campos se ha perdido	cuál de los 5 campos se ha pedido
626, t., 58	6080' FILE : TASTI	6080' FILE : TECLAS
647, A, 12	específico de tres valores)...	especifica de tres valores)...
772, A, 1	tracción se conseguirá por...	tracción estará constituida por...
772, A, 3	inmediatamente inferior se conseguirá...	inmediatamente inferior estará constituida...
788, A, 31	Bib-bang	Big-bang
806, C, t./g., 13	ción SUM (D6..D16).	ción SUM (D6...D16).
817, A, 11	COUNT (NS..NE).	COUNT (NS...NE).
817, B, 5	SUM (NS..NE).	SUM (NS...NE).
817, B, 9	ción es inmediato: SUM(C2..C5)...	ción es inmediato: SUM(C2...C5)...
817, B, 13, 14	$\frac{\text{SUM}(A1..A9)}{\text{COUNT}(A1..A9)}$	$\frac{\text{SUM}(A1...A9)}{\text{COUNT}(A1...A9)}$
818, A, 1	MAX (NS..NE), MIN (NS..NE).	MAX (NS...NE), MIN (NS...NE).
818, A, 4	NPV (S,NS..NE).	NPV (S,NS...NE).
818, B, 10	LOOKUP (M,NS..NE).	LOOKUP (M,NS...NE).
818, B, 11	rango NS..NE. ...	rango NS...NE. ...
818, t./g.	@COUNT (A2..A5) @SUM (C2..C5) @SUM (D2..D5) @SUM (E2..E5)	@COUNT (A2...A5) @SUM (C2...C5) @SUM (D2...D5) @SUM (E2...E5)
819, B, t./g., 7	se han aplicado. Así, en B14	se han aplicado. Así, en B12
819, B, t./g., 9	de la columna B y en D14...	de la columna B y en D12...
823, t./g., 11	SUM (B2..B6)	SUM (B2...B6)
823, t./g., 12	SUM (C2..C6)	SUM (C2...C6)
824, A, t./g., 10	SUM (B2..B6),...	SUM (B2...B6),...
826, D, t./g.	@SUM (E4..G4) @SUM (E5..G5) @SUM (E6..G6) @SUM (E7..G7) @SUM (E8..G8) @SUM (E9..G9)	@SUM (E4...G4) @SUM (E5...G5) @SUM (E6...G6) @SUM (E7...G7) @SUM (E8...G8) @SUM (E9...G9)
838, B, t./g., 7	cripción, columna D). ...	cripción, columna C). ...
968, B, 8	BLOCK CONTAINS 3 RECORDS	BLOCK CONTAINS 3 RECORDS.
968, B, 12 a 15	BLOCK CONTAINS 10 RECORDS BLOCK CONTAINS 1 TO 10 RECORDS BLOCK CONTAINS 100 CHARACTERS BLOCK CONTAINS 10 TO 900 CHARACTERS	BLOCK CONTAINS 10 RECORDS. BLOCK CONTAINS 1 TO 10 RECORDS. BLOCK CONTAINS 100 CHARACTERS. BLOCK CONTAINS 10 TO 900 CHARACTERS.
968, B, 22	BLOCK CONTAINS 1 RECORDS	BLOCK CONTAINS 1 RECORDS.

969, A, 10
 975, t./g., 6
 975, A, 4
 978, A, 2
 980, A, 4
 980, t./g., 18
 995, A, 12
 997, A, 38
 1000, A, 4
 1000, B, 5
 1000, B, 10
 1001, A, 4
 1001, A, 5
 1001, B,
 1002, A, 2
 1002, A, 7
 1003, A, 21
 1003, B, 28
 1004, A, 32
 1004, B, 33
 1024, A, 7
 1027, A, 5
 1027, A, 33
 1027, B, 34
 1027, A, 42
 1027, A, 45
 1027, B, 2
 1027, B, 5
 1027, B, 8
 1030, A, 2
 1092, t./g.
 1239, A, 21
 1353, B, 12
 1353, B, 13 a 20

LABEL RECORD IS OMITED
 Código-1: 33 caracteres
 33 caracteres, ...
 A X 9 P X * \$ B 0 + - . , / S V C R D B
 Inserta una coma...
 1234 9(4) \$1234
 SELECT FILE-A ASSIGN TO DISC FILE-A
 reabrirlo... (CLOSE, F1-
OPEN EXTEND nombre-de-file
 FILE 5
 FILE 5
READ nombre-de-file INTO nombre-de-dato
AT END frase-imperativa
 DISPLAY 'FIN FILE-A'
 UPON PRINTER
 CLOSE FILE-A
 WRITE nombre-de-record
 dato
 La condición... inalcanzable el
 AFTER ADVANCING 3 LINES.
 Cláusulas para el salto de páginas.
 [LINES AT BOTTOM margen-inferior]
 DIVIDE 10 BY 4 GIVING COCIENTE
 01 FICHA PIC X(3).
 MOVE SPACES TO AUXILIAR-FICHA
 MOVE FICHA TO AUXILIAR-FICHA
 MOVE SPACES TO AUXILIAR-FICHA
 MOVE FICHA TO AUXILIAR-FICHA
 TO DESCRIPTION
 MOVE 'ABC' TO TIPO-FICHA
 MOVE 83 TO AÑO
 MOVE NUMERO TO CAMPO

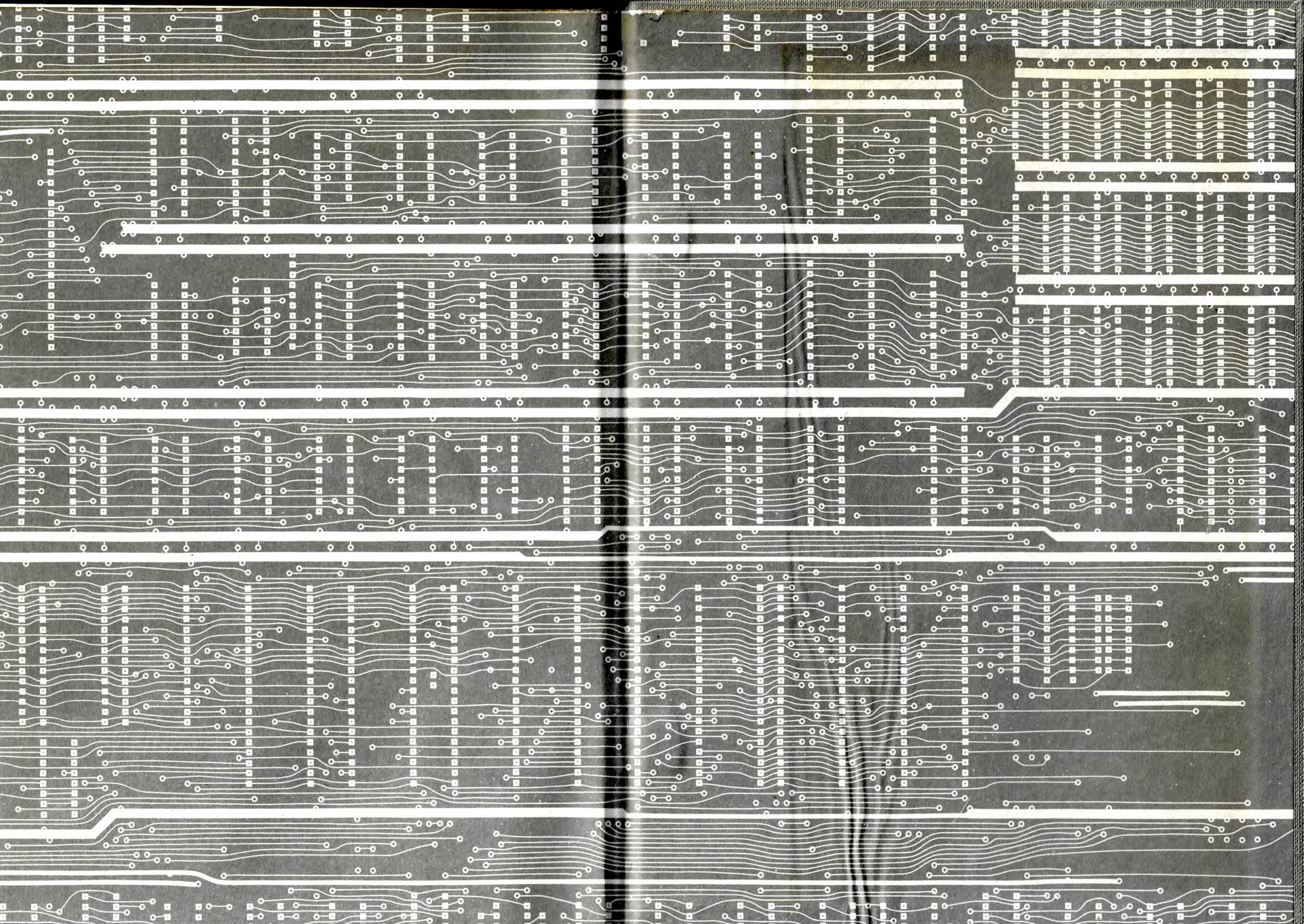
Tabla totalizadores mensuales														
1	2	3	4	5	6	7	8	9	10	11	12			

 READ Y READLN
 tante complejas y se emplea...
 trumentos electrónicos de medida) a cortas distancias (pocos metros). El protocolo es el tipo «byte serie, bit paralelo» y permite conversar a un número máximo de 15 instrumentos a alta velocidad (hasta un millón de caracteres por segundo).

LABEL RECORD IS OMITED.
 Código-1: 23 caracteres
 23 caracteres, ...
 A 9 P X * \$ B 0 + - . , / S V C R D B
 , Inserta una coma...
 1234 \$9(4) \$1234
 SELECT FILE-A ASSIGN TO DISC FILE-A.
 reabrirlo... (CLOSE F1-
OPEN EXTEND nombre-de-file.
 FILE 5.
 FILE 5.
READ nombre-de-file INTO nombre-de-dato.
AT END frase-imperativa.
 DISPLAY 'FIN FILE-A'
 UPON PRINTER.
 CLOSE FILE-A.
 WRITE nombre-de-record.
 dato.
 La condición... inaccesible el
 BEFORE ADVANCING 3 LINES.
 Cláusulas para el salto de página.
 [LINES AT BOTTOM margen-inferior].
 DIVIDE 10 BY 4 GIVING COCIENTE.
 01 FICHA.
 MOVE SPACES TO AUXILIAR-FICHA.
 MOVE FICHA TO AUXILIAR-FICHA.
 MOVE SPACES TO AUXILIAR-FICHA.
 MOVE FICHA TO AUXILIAR-FICHA.
 TO DESCRIPTION.
 MOVE 'ABC' TO TIPO-FICHA.
 MOVE 83 TO AÑO.
 MOVE NUMERO TO CAMPO.

Tabla totalizadores mensuales														
1	2	3	4	5	6	7	8	9	10	11	12			

 READ y READLN
 tante complejas.
El protocolo IEEE 488
 En 1978, el Instituto de estandarización lo formalizó, y se emplea para controlar periféricos rápidos a cortas distancias. El protocolo es del tipo «byte serie, bit paralelo» y permite colokuar a alta velocidad un número máximo de 15 aparatos.

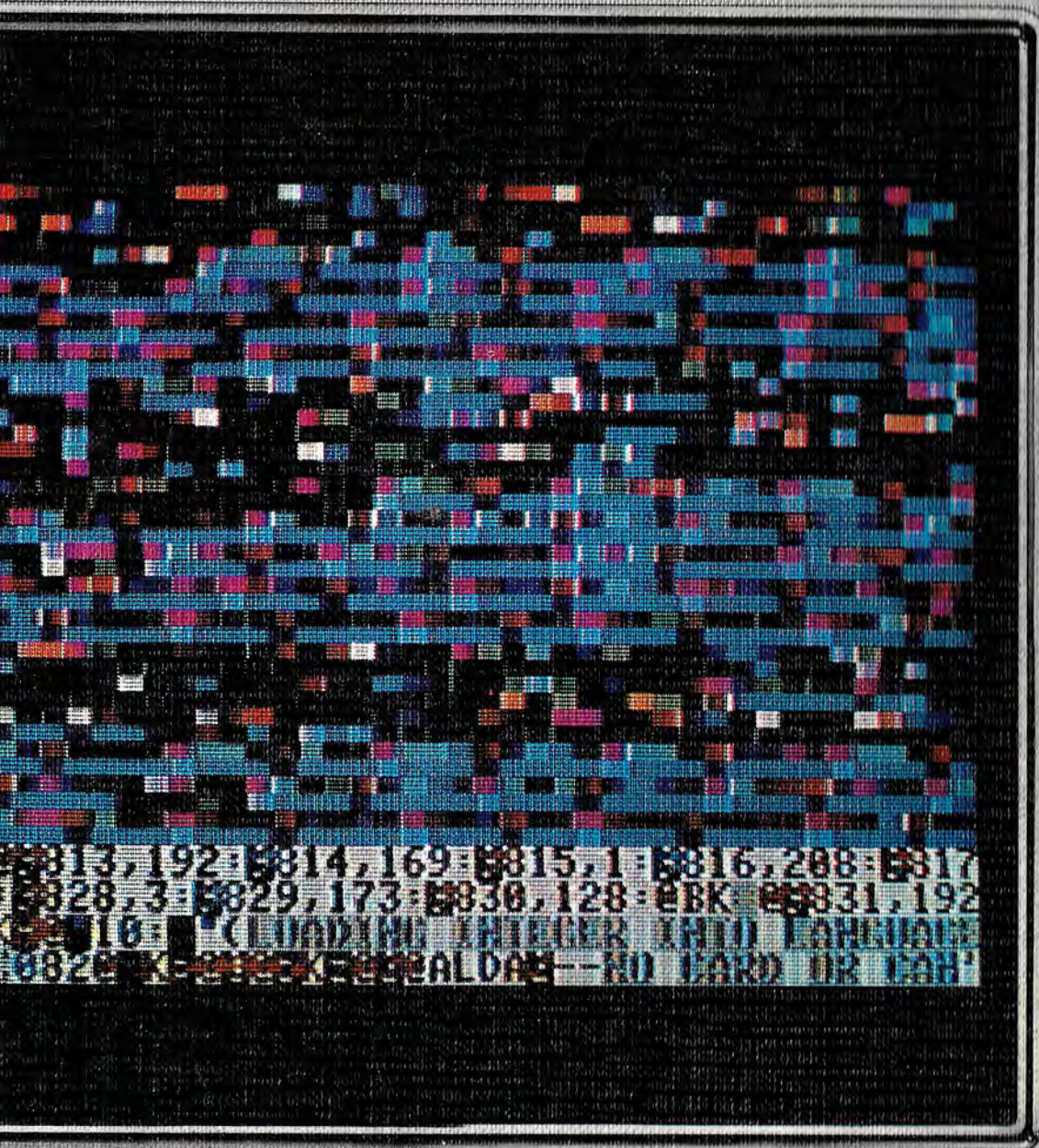


EDICIONES
FORUM



BASIC

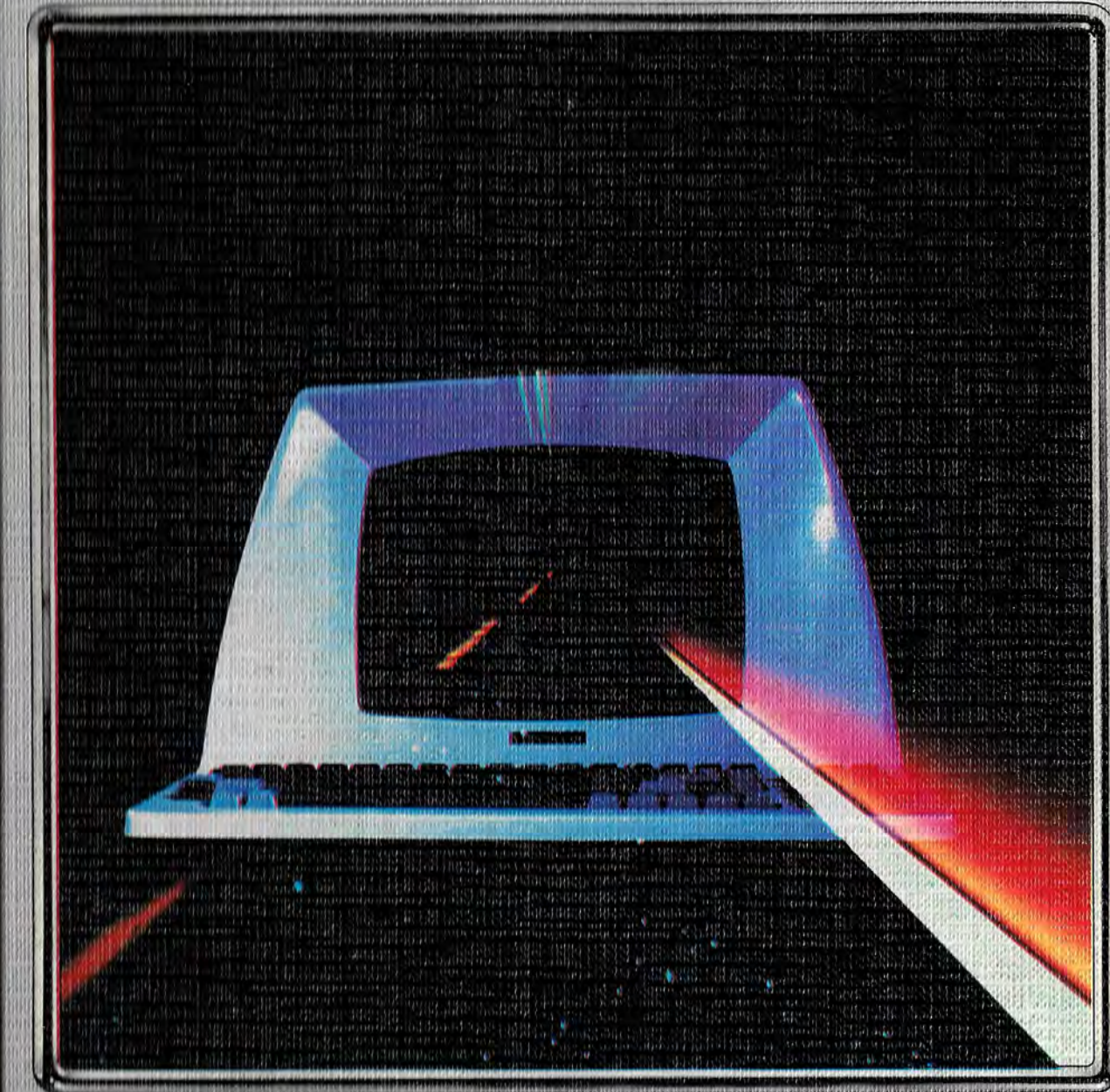
ENCICLOPEDIA DE LA INFORMATICA
MINIORDENADORES Y ORDENADORES PERSONALES



813, 192 814, 169 815, 1 816, 288 817
828, 3 829, 173 830, 128 BK 831, 192
18: CLIPPING THICKER XHD LAMINAR
828 X 200 KEALDAN NO DARD OR DAN



ENCICLOPEDIA DE LA INFORMATICA
MINIORDENADORES Y ORDENADORES PERSONALES



EDICIONES FORUM