

Informática 24 Y PROGRAMACIÓN

PASO A PASO



PROGRAMAS EDUCATIVOS
PROGRAMAS DE UTILIDAD
PROGRAMAS DE GESTION
PROGRAMAS DE JUEGOS

▼ BASIC ▼ MAQUINA ▼ PASCAL ▼ LOGO ▼ OTROS LENGUAJES ▼
▼ TECNICAS DE ANALISIS Y DE PROGRAMACION ▼

Informática y programación

24

PASO A PASO



PROGRAMAS EDUCATIVOS
PROGRAMAS DE UTILIDAD
PROGRAMAS DE GESTION
PROGRAMAS DE JUEGOS

▼ BASIC ▼ MAQUINA ▼ PASCAL ▼ LOGO ▼ OTROS LENGUAJES ▼
▼ TECNICAS DE ANALISIS Y DE PROGRAMACION ▼

▼ EDICIONES ▼ SIGLO ▼ CULTURAL ▼

Una publicación de

EDICIONES SIGLO CULTURAL, S.A.

Director-editor:

RICARDO ESPAÑOL CRESPO.

Gerente:

ANTONIO G. CUERPO.

Directora de producción:

MARIA LUISA SUAREZ PEREZ.

Directores de la colección:

MANUEL ALFONSECA, Doctor Ingeniero de Telecomunicación
y Licenciado en Informática.

JOSE ARTECHE, Ingeniero de Telecomunicación.

Diseño y maquetación:

BRAVO-LOFISH.

Fotografía:

EQUIPO GALATA.

Dibujos:

JOSE OCHOA

TECNICAS DE PROGRAMACION: Manuel Alfonseca, Doctor Ingeniero de Telecomunicación y Licenciado en Informática. TECNICAS DE ANALISIS: José Arteché, Ingeniero en Telecomunicación. LENGUAJE MAQUINA 8086: Juan Rojas Licenciado en Ciencias Físicas e Ingeniero Industrial. PASCAL: Juan Ignacio Puyol, Ingeniero Industrial. PROGRAMAS (educativos, de utilidad, de gestión y de juegos): Francisco Morales, Técnico en Informática y colaboradores. Coordinador de AULA DE INFORMATICA APLICADA (AIA): Alejandro Marcos, Licenciado en Ciencias Químicas. BASIC: Esther Maldonado, Diplomada en Arquitectura. INFORMATICA BASICA: Virginia Muñoz, Diplomada en Informática. LENGUAJE MAQUINA Z-80: Joaquín Salvachúa, Diplomado en Telecomunicación y José Luis Tojo, Diplomado en Telecomunicación. LENGUAJE MAQUINA 6502: (desde el tomo 5): Juan José Gómez, Licenciado en Química. LOGO: Cristina Manzanera, Licenciada en Informática. APLICACIONES: Sociedad Tamariz, Diplomada en Telecomunicación. OTROS LENGUAJES (COBOL): Eloy Pérez, Licenciado en Informática. Ana Pastor, Licenciada en Informática.

Ediciones Siglo Cultural, S.A.

Dirección, redacción y administración:

Pedro Teixeira, 8, 2.ª planta. Teléf. 810 52 13. 28020 Madrid.

Publicidad:

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

Distribución en España:

COEDIS, S.A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.

Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:

CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.

Buenos Aires - 1.290. Argentina.

Todos los derechos reservados. Este libro no puede ser, en parte o totalmente, reproducido, memorizado en sistemas de archivo, o transmitido en cualquier forma o medio, electrónico, mecánico, fotocopia o cualquier otro, sin la previa autorización del editor.

ISBN del tomo: 84-7688-161-4

ISBN de la obra: 84-7688-068-7

Fotocomposición:

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

Imprime:

MATEU CROMO. Pinto (Madrid).

© Ediciones Siglo Cultural, S.A., 1987.

Depósito legal: M-5-677-1987

Printed in Spain - Impreso en España.

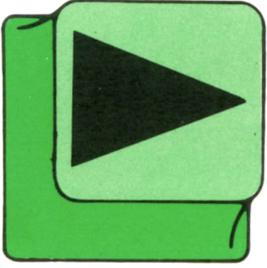
Suscripciones y números atrasados:

Ediciones Siglo Cultural, S.A.

Pedro Teixeira, 8, 2.ª planta. Teléf. 810 52 13. 28020 Madrid.

Agosto, 1987.

P.V.P. Canarias: 335,-.



INDICE

4	INFORMATICA BASICA
8	MAQUINA Z-80
11	PROGRAMAS EDUCATIVOS PROGRAMAS DE UTILIDAD PROGRAMAS DE GESTION PROGRAMAS DE JUEGOS
26	TECNICAS DE ANALISIS
28	TECNICAS DE PROGRAMACION
32	APLICACIONES
35	PASCAL
39	OTROS LENGUAJES

INFORMATICA BASICA

DISPOSITIVOS DE ENTRADA-SALIDA

Introducción

Los dispositivos periféricos son utilizados por la CPU (unidad central de proceso), como elementos de intercambio de información con el exterior.

Debe tenerse en cuenta que hay gran variedad de dispositivos y soportes de información, por lo que también han de ser variados los conductos por los que se transmite.

Podemos agrupar los periféricos en tres grupos: dispositivos de almacenamiento, dispositivos de entrada y dispositivos de salida. Algunas veces los mismos dispositivos de entrada pueden funcionar para salida.

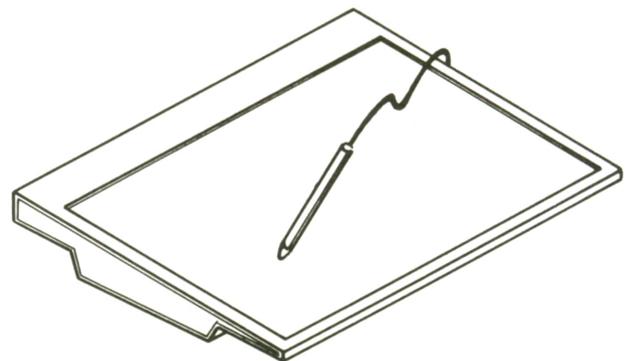
Dispositivos de entrada y salida

Como su nombre indica, son los dispositivos que permitirán la entrada y salida de la información del y al núcleo del ordenador.

Podemos encontrar dentro del grupo de periféricos de entrada teclados, lectoras de tarjetas, lectoras de cintas, lectoras ópticas, lectoras magnéticas, etc. Mientras que una impresora, un plotter, un monitor, etc., son dispositivos periféricos de salida. Cada uno de estos elementos utiliza un soporte propio para asumir y transferir la información. Así, un teclado utilizará las pulsaciones que el operador ejecute para transmitir la información; un lector óptico examinará los caracteres impresos; una impresora nos devolverá la información procesada, escrita en un papel, etc.

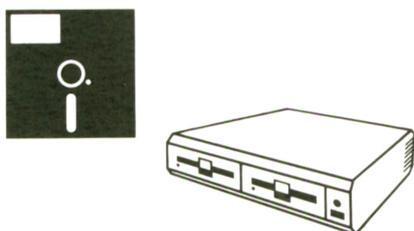


Relación entre los dispositivos que componen la parte física de la máquina.



El tablero gráfico permite introducir datos gráficos, por tanto, es un dispositivo de entrada de datos.

Los periféricos han sufrido un gran desarrollo en cuanto a su volumen; en un principio predominaron las lectoras de tarjetas perforadas; pero hoy en día éstos han dejado paso a los terminales vídeo, impresoras láser, incluso puede hablarse de sintetizadores y reconocedores de voz. En la actualidad, los dispositivos de entrada y salida más utilizados, además de las impresoras, son los terminales de vídeo.



 *Los disquetes son dispositivos que sirven para conservar datos o programas que se pueden volver a utilizar.*

Periféricos de almacenamiento

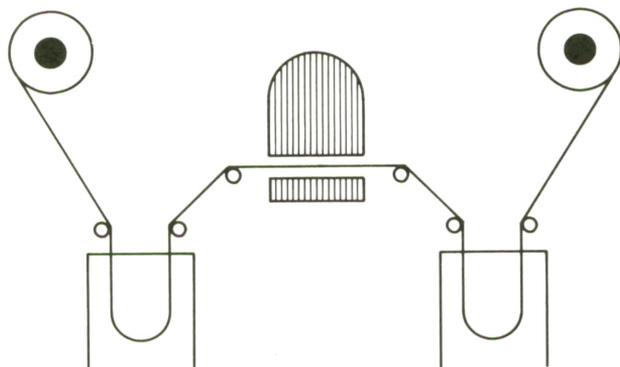
Este tipo de dispositivos tienen una gran importancia, ya que han sido los principales medios de desarrollo del ordenador actual. No son otra cosa que dispositivos capaces de almacenar información, es decir, son memorias.

Tan sólo mencionaremos los dos tipos más importantes: la banda o cinta magnética y el disco. Son elementos magnéticos todos ellos y su principio de funcionamiento es básicamente el mismo.

La música, el sonido y las imágenes pueden ser almacenadas en cintas y discos. Esta se trata de información almacenada y recuperable tantas veces como se quiera. La diferencia entre la información almacenada por ordenador y la de disco está en que la primera utilizará técnicas digitales y la segunda analógicas (aunque hoy en día tienden a ser digitales también).

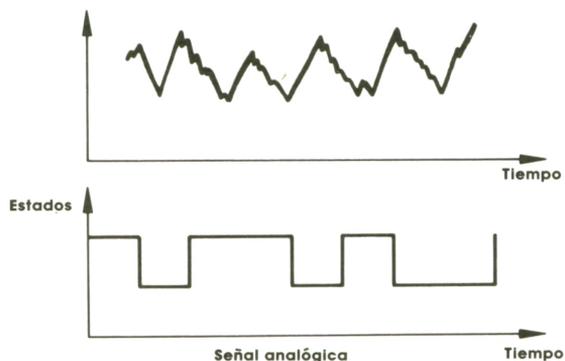
La diferencia entre las señales analógicas y las digitales están básicamente en que la señal analógica maneja un espectro más o menos amplio de intensidades y/o frecuencias; mientras que la di-

gital sólo maneja dos posibles estados 1 y 0. A la hora de grabar estas señales, las digitales, por distinguir dos estados, son mucho más fáciles de plasmar en un medio magnético. Sin embargo, la señal analógica se almacenará en base a provocar diferencias en la densidad de las partículas magnetizables situadas en una cinta cassette, por ejemplo, o bien creando una serie de surcos en la superficie de un disco.



 *Señal digital.*

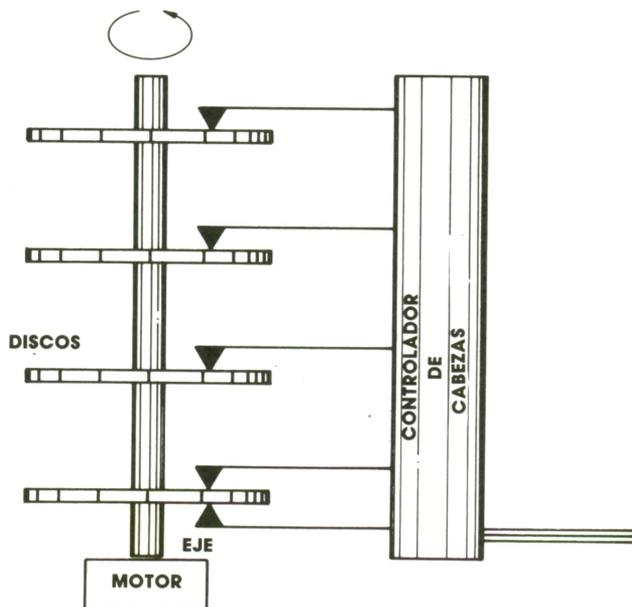
La cinta magnética es de material plástico y está recubierta en una de sus dos caras por una película delgada de material magnetizable. Las cintas más típicas tienen media pulgada de ancho y 1200 ó 2400 pies de longitud. Está enrollada en un carrete y se va desenrollando pasando a otro carrete. Mientras esto sucede, desfila frente a unas cabezas lectoras. Puede moverse hacia adelante o hacia atrás.



 *Representación esquemática de una cinta magnética.*

Los discos son superficies planas circulares de dos caras. Sobre éstos se deposita una película de material magnetizable. El disco gira alrededor de un eje que

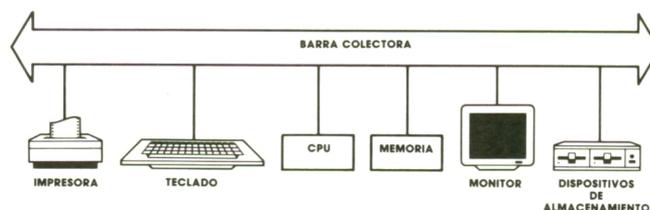
pasa por el centro del círculo y es perpendicular a su plano. La cabeza de lectura-escritura se sitúa sobre él. Graba sobre el disco a base de circunferencias concéntricas que se llaman **pistas**. Cada pista lleva la misma cantidad de información, por lo que las pistas interiores tendrán menos densidad de grabación. La cabeza lectora puede acceder a cualquier pista que queramos sin tener que pasar por los anteriores.



 Representación de una unidad de cuatro discos; las cabezas pueden leer los discos independientemente.

Canales de comunicación

Estos se sitúan entre la CPU y los periféricos. Son los medios por los cuales la CPU puede comunicarse con los periféricos. En la actualidad se tiende a la estandarización de éstos para evitar problemas entre diferentes ordenadores que se quieren conectar entre sí.



 Disposición de la barra colectora para el control de la comunicación entre los diferentes componentes del hardware.

Básicamente constan de:

- Cable y conectores.
- Corrientes, impedancias y tensiones. Elementos electrónicos.
- Estructura de la señal. Elemento lógico.

Hay dos tipos principales, en serie y en paralelo. Los que tienen configuración en serie están formados por dos hilos de transmisión, uno para emisión y otro para recepción, además de un cierto número de señales dedicadas al diálogo de los elementos que une. Estos hacen la comunicación entre los dispositivos «bit a bit». Aunque estas líneas de emisión y recepción funcionan en serie, las de diálogo funcionan en paralelo.

El interface (como también se conoce a los canales de transmisión) en paralelo se distingue del de serie en que se efectúa la emisión y recepción simultánea de varios bits; generalmente un byte (8 bits). La transmisión es más rápida, pero implica un número mayor de cables.

Codificadores y decodificadores

Son circuitos lógicos con un gran número de entradas que proporcionan a la salida una combinación binaria determinada por la entrada activa en ese momento. Estos circuitos ayudan a crear una distribución de tareas de comunicación más organizadas. Cuando un dispositivo comienza a crecer en recursos el cableado o circuitería de comunicación es el elemento más directamente responsable de acumulaciones de energía y, por tanto, de calor, y del aumento de los tiempos de ejecución y control de la máquina. Por esta razón en los sistemas grandes se suelen interconectar los elementos tratando de evitar un crecimiento excesivo.

Un esquema muy usado es el de la barra colectora. A ésta se le suele asignar un protocolo de comunicación propio creado por los codificadores y decodificadores que proporciona que a cada elemento le llegue la información de la forma deseada, y evitando que las uni-

dades conectadas a la barra tengan comunicación entre sí, sin que se produzcan interferencias. Con este tipo de esquema, los elementos de la conmutación se minimizan porque cada elemento del sistema sólo necesita la conexión con la barra.

MAQUINA Z-80

SPECTRUM, AMSTRAD, MSX

Programas de ejemplo

RATAREMOS en este capítulo de aclarar las ideas del lector ya que probablemente, y después de una descripción del ensamblador tan densa, no hayan queda-

do claras las posibles aplicaciones de algunas instrucciones.

Para ello empezaremos realizando un sencillo programa para multiplicar 2 bytes, y luego iremos haciendo otros ejemplos, cada vez más complicados. De todas formas, en el tomo 39 se tratarán nuevos ejemplos de utilidad para los ordenadores que dispongan del microprocesador Z-80.

Con ello esperamos que el lector tome idea clara acerca de las dos características principales del lenguaje máquina: su potencia y su dificultad. No debe olvidarse que, como en todos los lenguajes, no se llega a una utilización óptima de sus posibilidades sin haber adquirido mucha práctica y gran cantidad de conocimientos acerca del entorno en que el lenguaje se utilizará.

En nuestro caso, el lenguaje máquina del Z-80 será la máquina a nivel hardware. Sin un conocimiento claro acerca de su arquitectura será muy difícil obtener la máxima potencia de la misma.

Primer ejemplo. Multiplicación de dos bytes

La utilidad de este ejemplo es mostrar la lógica binaria y cómo se realizan con

ella operaciones complicadas. Veamos el proceso del programa con un sencillo organigrama.

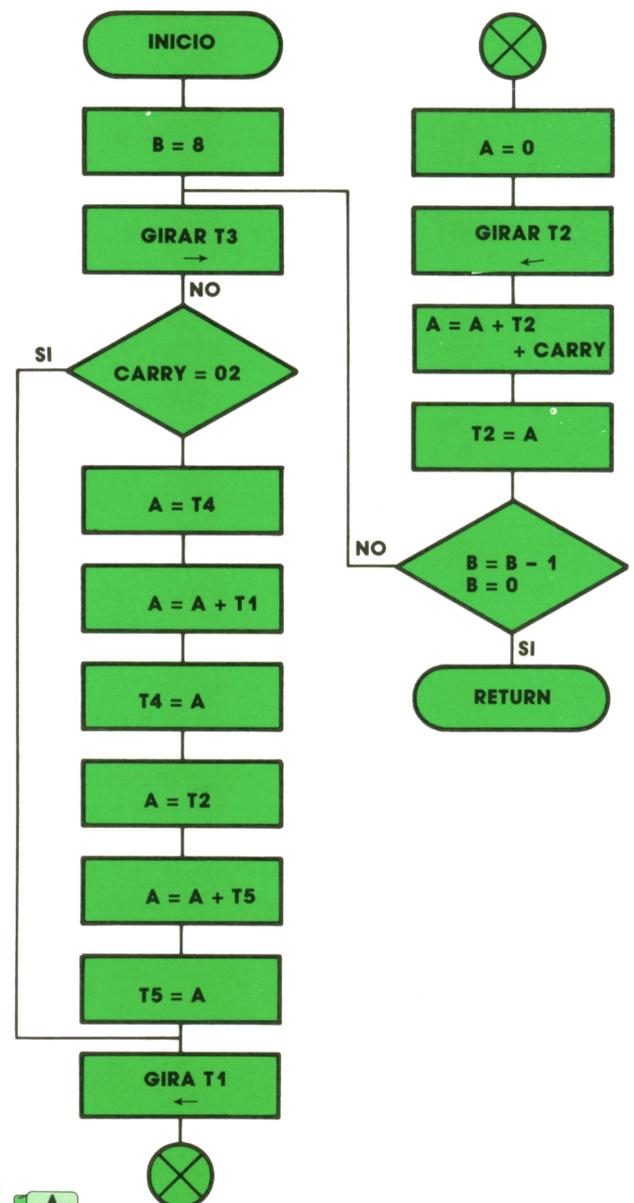


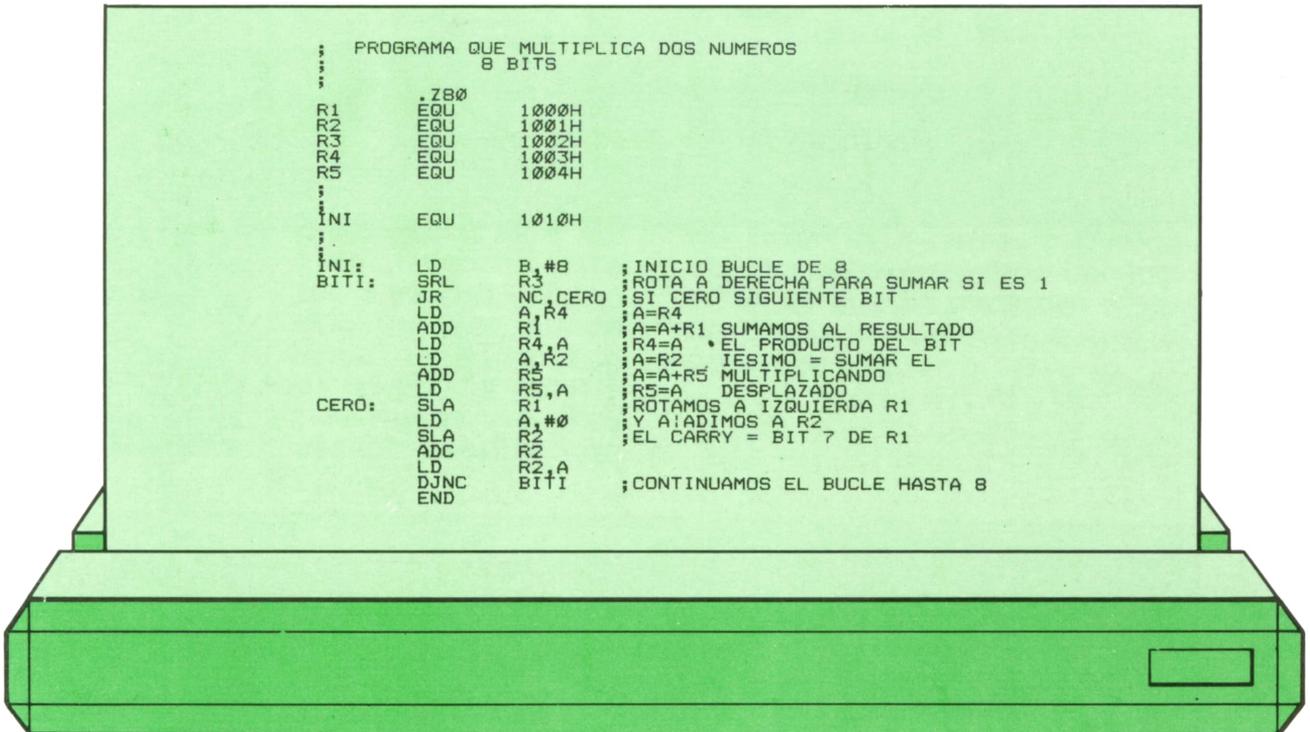
Fig. 9.1

Explicaremos brevemente el proceso:

Se trata de multiplicar dos números binarios. Lo haremos como si multiplicásemos dos números a mano; multiplicamos un dígito del multiplicador por todo el multiplicando y sumamos el resultado al producto, desplazando previamente, dependiendo del orden del dígito que multiplique. Si observamos detenidamen-

te, al utilizar solamente los números 1 y 0, el resultado de multiplicar es el mismo número o cero, respectivamente. Por ello, la operación multiplicar se reduce a sumar números desplazados.

En el listado en ensamblador los comentarios explican paso a paso cuál es el proceso.

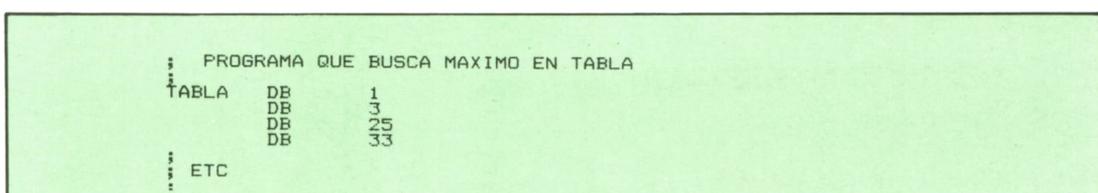


El algoritmo es el siguiente. Desplazamos el byte multiplicador poniendo el dígito que va a multiplicar en el carry. Si es cero, pasamos al siguiente dígito, desplazando el multiplicando (T1) con ayuda del registro adicional T2, para que en sucesivas sumas el par T2T1 contenga ya el número desplazado convenientemente. Si el carry no es cero, sumamos el par T2T1 al resultado (la multiplicación por 1 es obvia) y luego continuamos el proceso como si el dígito que multiplica fuese cero; es decir, desplazamos el par T2T1, etcétera.

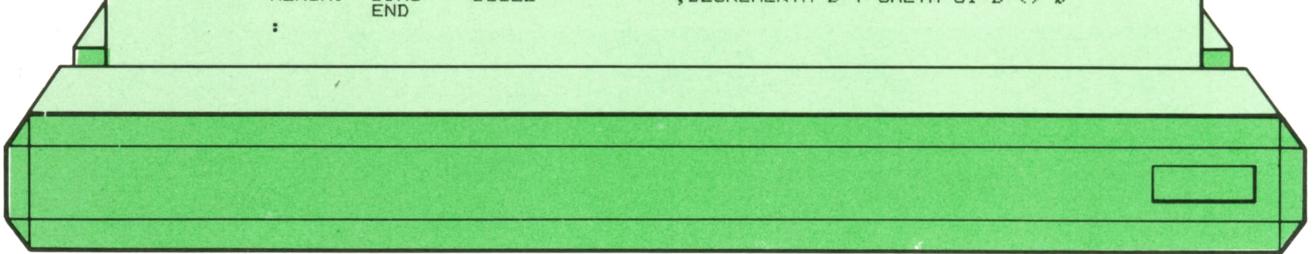
Realizamos todo el proceso ocho veces (reguladas por el contenido del registro B), que corresponden a los 8 bits del byte multiplicador T3.

Segundo ejemplo. Búsqueda del máximo de una tabla

Este ejemplo es muy sencillo. Recorremos una tabla de bytes buscando el que tenga el valor mayor. A continuación puede verse el listado comentando paso a paso el proceso.



```
LTABLA DB      4
:
:
:
BUSCA EQU      1010H
:
BUSCA: LD      HL, TABLA      ; HL = PRINCIPIO TABLA
LD      B, LTABLA-1        ; B = NUMERO COMPARACIONES
LD      A, (HL)            ; PRIMER VALOR
LD      D, A               ; TOMADO COMO MAXIMO
BUCLE: INC     HL           ; COGER SIGUIENTE
CP      D                   ; COMPARARLO CON MAXIMO
JM      M, MENOR           ; SALTA SI A < D
LD      D, A               ; ACTUALIZA MAXIMO
MENOR: DJNZ   BUCLE        ; DECREMENTA B Y SALTA SI B <> 0
:
:
```



PROGRAMAS

EDUCATIVOS • DE UTILIDAD • DE GESTION • DE JUEGOS



Programa:
Editor

de textos
para SPECTRUM

L programa que vamos a ver en este tomo es una utilidad que todo el mundo suele necesitar. Este es un EDITOR DE TEXTOS. Nos servirá para escribir nuestras car-

tas, apuntes, informes o cualquier cosa que nosotros deseemos, con la particularidad de que en cualquier momento podemos modificar y/o borrar cualquier parte del texto.

```

10 REM *****
20 REM * EDITOR DE TEXTOS *
30 REM *****
40 REM
50 REM *****
60 REM *(c)Ed. Siglo Cultural*
70 REM *****
80 REM
90 PAPER 0
100 BORDER 0
110 INK 6
120 CLEAR 29000
130 PRINT AT 10,9; FLASH 1;"CARGANDO DATAS"
140 PRINT AT 12,7; FLASH 1; INVERSE 1; INK 4;"ESPERE UN MOMENTO."
160 LET CHEK=0
170 LET LIN=7000
180 LET COL=0
182 LET X1=29500
184 LET X2=2850
186 LET SW=0
188 LET NN=17
190 FOR I=X1 TO X1+X2
200   READ A
210   LET COL=COL+1
220   IF COL=NN THEN LET NUM=A: IF NUM<>CHEK THEN CLS : PRINT "ERROR DE DATA
EN LA LINEA ";LIN: GO TO 9999
230   IF COL=NN THEN LET COL=0: LET LIN=LIN+2: LET CHEK=0: LET I=I-1: GO TO 2
60
240   POKE I, A
250   LET CHEK=CHEK+A
260 NEXT I
270 READ A
300 IF SW=0 THEN LET SW=1: LET NN=9: LET CHEK=0: LET COL=0: LET LIN=9000: LET
X1=32768: LET X2=1024: GO TO 190
310 CLS

```

```

320 PRINT " PROGRAMA CARGADO EN MEMORIA."
330 PRINT : PRINT : PRINT
340 PRINT " QUIERES GUARDARLO? (S/N)"
350 LET A$=INKEY$
360 IF A$="" THEN GO TO 350
370 IF A$="N" OR A$="n" THEN GO TO 430
380 IF A$<>"S" AND A$<>"s" THEN GO TO 350
390 PRINT : PRINT : PRINT
400 PRINT " PREPARA EL CASSETTE"
410 SAVE "CODIGO"CODE 29500,3000
420 SAVE "SET"CODE 32768,1024
430 PRINT : PRINT : PRINT
440 PRINT " QUIERES EJECUTARLO? (S/N)"
450 PAUSE 2: PAUSE 2:- LET A$=INKEY$
460 IF A$="" THEN GO TO 450
470 IF A$="N" OR A$="n" THEN GO TO 9999
480 IF A$<>"S" AND A$<>"s" THEN GO TO 450
490 RANDOMIZE USR 29500
9999 PRINT : PRINT "ADIOS ..."

```

El editor de textos se compone de tres programas distintos. El primero de ellos es el que transforma una serie de líneas de datos en un fichero de bytes y lo almacena en una cinta de cassette. Introduce este programa teniendo mucho cuidado de

las líneas DATA y lo ejecuta. Una vez que el ordenador haya almacenado todo el código máquina en la memoria te dirá que prepares una cinta de cassette para su grabación.

```

10 REM *****
20 REM * EDITOR DE TEXTOS *
30 REM *****
40 REM
50 REM *****
60 REM *(c)Ed. Siglo Cultural*
70 REM *****
80 REM
90 PAPER 0
100 BORDER 0
110 INK 6
120 CLEAR 29000
130 PRINT AT 10,13; FLASH 1;"ESPERE"
140 PRINT AT 12,4; FLASH 1; INVERSE 1; INK 4;"CARGANDO CODIGO MAQUINA."
150 LOAD "CODIGO"CODE
160 LOAD "SET"CODE
170 RANDOMIZE USR 29500

```

Una vez grabado el código máquina tenemos que introducir el segundo programa. Este sólo se encargará de leer el

código máquina que hemos almacenado en cinta y ejecutarlo.

```

1000 ;*****
1001 ;*      *
1002 ;*  E D I T O R  *
1003 ;*      D E      *
1004 ;*  T E X T O S  *
1005 ;*      *
1006 ;*****
1007 ;
1008      ORG 29500
1009      LD A,128
1010      LD (23607),A
1011      LD A,8
1012      LD (23658),A
1013      CALL SET_MD
1014      CALL SET_MI
1015      LD A,1
1016      LD (23617),A
1017      CALL EXT_M
1018      LD A,1
1019      CALL 5633
1020      CALL OVER_1
1021      LD HL,LOWER
1022      LD A,(CY)
1023      PUSH AF
1024      XOR A
1025      LD (CY),A
1026      CALL LINEA
1027      POP AF
1028      LD (CY),A
1029      CALL DOS
1030      LD A,(MI)
1031      LD (CX),A
1032      CALL CP_L
1033      CALL HL_COR
1034      LD B,65
1035 R_SP  PUSH BC
1036      LD B,255
1037 R_SPC LD (HL),32
1038      INC HL
1039      DJNZ R_SPC
1040      POP BC
1041      DJNZ R_SP
1042      JR P_COLI
1043 HL_COR LD HL,35000
1044      LD A,(CX)
1045      LD D,0
1046      LD E,A
1047      PUSH DE
1048      LD A,(CY)
1049      AND A
1050      JR Z,HL_1
1051      LD DE,64
1052      LD B,A
1053 HL_0  ADD HL,DE
1054      DJNZ HL_0
1055 HL_1  POP DE
1056      ADD HL,DE
1057      RET
1058 AG   CALL KEY
1059      AND A
1060      PUSH AF
1061      CALL NZ,KEY_P
1062      CALL CURSOR
1063      CALL CURSOR
1064      POP AF
1065      JR Z,AG
1066 P_COLI LD A,(CX)
1067      LD (COORDS),A

```

```

1068      CALL UNO
1069      LD A,22
1070      RST 16
1071      LD A,1
1072      RST 16
1073      XOR A
1074      RST 16
1075      CALL P_NCOL
1076      LD A,(CY)
1077      LD (COORDS),A
1078      CALL UNO
1079      LD A,22
1080      RST 16
1081      LD A,1
1082      RST 16
1083      LD A,4
1084      RST 16
1085      CALL P_NCOL
1086      JR AG
1087 KEY   XOR A
1088      LD (23560),A
1089      RST 56
1090      LD A,(23560)
1091      LD (LAST_K),A
1092      RET
1093 KEY_P CP 128
1094      RET NC
1095      CP 13
1096      JP Z,ENTER
1097      CP 12
1098      JP Z,DELETE
1099      CP 24
1100      JP Z,DELETE
1101      CP 6
1102      JP Z,CP_L
1103      CP 26
1104      JP Z,CP_L
1105      CP 15
1106      JP Z,CINTA
1107      CP 1
1108      JP Z,CINTA
1109      CP 8
1110      JP Z,CUR_L
1111      CP 9
1112      JP Z,CUR_R
1113      CP 10
1114      JP Z,CUR_D
1115      CP 11
1116      JP Z,CUR_U
1117      CP 7
1118      JP Z,INSTR
1119      CP 25
1120      JP Z,INSTR
1121      CP 5
1122      JP Z,MARG_D
1123      CP 30
1124      RET Z
1125      CP 29
1126      RET Z
1127      CP 31
1128      RET Z
1129      CP 28
1130      JP Z,MARG_D
1131      CP 4
1132      JP Z,MARG_I
1133      CP 27
1134      JP Z,MARG_I
1135      CP 14
1136      JP Z,EXT_M

```

PROGRAMAS

```

1137 CP 16
1138 JP Z, DE_LI
1139 CP 17
1140 JP Z, LI_IZ
1141 CP 18
1142 JP Z, LI_DE
1143 CP 22
1144 JP Z, CU_FI
1145 CP 23
1146 JP Z, CU_PR
1147 CP 19
1148 JP Z, IN_LI
1149 CP 20
1150 JP Z, IN_CH
1151 CP 21
1152 JP Z, CE_LI
1153 CP 2
1154 JP Z, BO_CH
1155 CP 3
1156 JP Z, IMPRES
1157 CALL BEEP
1158 CALL OVER_1
1159 CALL AT
1160 LD A, (LAST_K)
1161 PUSH AF
1162 CALL CL_CHR
1163 POP AF
1164 LD (LAST_K), A
1165 PUSH AF
1166 CALL HL_COR
1167 POP AF
1168 LD (HL), A
1169 CALL AT
1170 LD A, (CX)
1171 SRL A
1172 JR C, DER
1173 CALL NC, IZQ
1174 JR SET_P
1175 DER LD A, (LAST_K)
1176 RST 16
1177 JR SET_P
1178 IZQ LD HL, (D_SET)
1179 LD D, 0
1180 LD A, (LAST_K)
1181 LD E, A
1182 LD B, 8
1183 AG_1 ADD HL, DE
1184 DJNZ AG_1
1185 LD D, H
1186 LD E, L
1187 LD B, 8
1188 RI LD A, (HL)
1189 SLA A
1190 SLA A
1191 SLA A
1192 SLA A
1193 LD (HL), A
1194 INC HL
1195 DJNZ RI
1196 LD A, (LAST_K)
1197 RST 16
1198 EX DE, HL
1199 LD B, 8
1200 RD LD A, (HL)
1201 SRL A
1202 SRL A
1203 SRL A
1204 SRL A
1205 LD (HL), A

```

```

1206 INC HL
1207 DJNZ RD
1208 RET
1209 SET_P CALL COMP_1
1210 LD A, (CX)
1211 INC A
1212 LD HL, MD
1213 CP (HL)
1214 JR C, SET_0
1215 JR NZ, NEW_L
1216 SET_0 CP 64
1217 JR Z, NEW_L
1218 LD (CX), A
1219 RET
1220 NEW_L LD A, (MI)
1221 LD (CX), A
1222 LD A, (CY)
1223 INC A
1224 LD (CY), A
1225 LD A, (C_Y)
1226 CP 21
1227 PUSH AF
1228 CALL Z, SCR_U
1229 POP AF
1230 RET Z
1231 INC A
1232 LD (C_Y), A
1233 RET
1234 IMPRES CALL S^
1235 LD A, 128
1236 LD (23607), A
1237 LD A, (CX)
1238 LD D, A
1239 LD A, (CY)
1240 LD E, A
1241 PUSH DE
1242 LD A, (C_Y)
1243 PUSH AF
1244 XOR A
1245 LD (C_Y), A
1246 LD (CX), A
1247 LD (CY), A
1248 LD B, 11
1249 IMPRE PUSH BC
1250 CALL SCREEN
1251 PUSH AF
1252 PUSH BC
1253 PUSH DE
1254 PUSH HL
1255 CALL 3756
1256 POP HL
1257 POP DE
C1258 POP BC
1259 POP AF
1260 LD A, (CY)
1261 ADD A, 22
1262 LD (CY), A
1263 POP BC
1264 DJNZ IMPRE
1265 LD A, 8
1266 LD (C_Y), A
1267 CALL SCREEN
1268 LD B, 112
1269 LD HL, #4800
1270 DI
1271 CALL 3762
1272 CALL S^^
1273 POP AF
1274 LD (C_Y), A

```

```

1275     POP  DE
1276     LD   A,D
1277     LD   (CX),A
1278     LD   A,E
1279     LD   (CY),A
1280     JP   BEEP
1281 BO_CH CALL HL_COR
1282     PUSH HL
1283     POP  DE
1284     INC  HL
1285     LD   A,(CX)
1286     LD   B,A
1287     LD   A,64
1288     SUB  B
1289     LD   B,0
1290     LD   C,A
1291     LDIR
1292     LD   A,32
1293     LD   (DE),A
1294     CALL LOP_
1295     CALL CL_LIN
1296     CALL P_L_C
1297     JP   BEEP
1298 IN_CH LD   A,(CY)
1299     PUSH AF
1300     LD   (CY),A
1301     LD   A,(CX)
1302     PUSH AF
1303     LD   A,64
1304     LD   (CX),A
1305     CALL HL_COR
1306     POP  AF
1307     LD   (CX),A
1308     POP  AF
1309     LD   (CY),A
1310     DEC  HL
1311     LD   A,(HL)
1312     CP   32
1313     RET  NZ
1314     EX  DE,HL
1315     PUSH DE
1316     POP  HL
1317     DEC  HL
1318     LD   A,(CX)
1319     LD   B,A
1320     LD   A,63
1321     SUB  B
1322     LD   C,A
1323     LD   B,0
1324     LDDR
1325     CALL BEEP
1326     CALL HL_COR
1327     LD   (HL),32
1328     LD   A,(CX)
1329     LD   D,A
1330     LD   A,(CY)
1331     LD   E,A
1332     PUSH DE
1333     LD   A,(C_Y)
1334     PUSH AF
1335     LD   B,1
1336     JP   P_L
1337 CE_LI LD   A,(CX)
1338     LD   D,A
1339     LD   A,(CY)
1340     LD   E,A
1341     PUSH DE
1342     LD   A,(C_Y)
1343     PUSH AF

```

```

1344     XOR  A
1345     LD   (CX),A
1346     CALL HL_COR
1347     LD   B,0
1348 CELI_I LD   A,(HL)
1349     INC  HL
1350     INC  B
1351     CP   32
1352     JR   Z,CELI_I
1353     PUSH BC
1354     LD   A,63
1355     LD   (CX),A
1356     CALL HL_COR
1357     LD   C,0
1358 CELI_D LD   A,(HL)
1359     DEC  HL
1360     INC  C
1361     CP   32
1362     JR   Z,CELI_D
1363     POP  DE
1364     LD   B,D
1365     DEC  B
1366     DEC  C
1367     LD   A,B
1368     ADD  A,C
1369     LD   D,A
1370     LD   A,64
1371     SUB  D
1372     PUSH AF
1373     XOR  A
1374     LD   (CX),A
1375     PUSH BC
1376     CALL HL_COR
1377     POP  BC
1378     LD   D,0
1379     LD   E,B
1380     ADD  HL,DE
1381     LD   DE,23350
1382     POP  AF
1383     PUSH BC
1384     LD   B,0
1385     LD   C,A
1386     PUSH BC
C1387     LDIR
1388     POP  DE
1389     POP  BC
1390     PUSH DE
1391     LD   A,B
1392     ADD  A,C
1393     SRL  A
1394     PUSH AF
1395     CALL HL_COR
1396     POP  AF
1397     LD   B,A
1398     PUSH AF
1399     CALL PON
1400     EX  DE,HL
1401     LD   HL,23350
1402     POP  AF
1403     POP  BC
1404     LDIR
1405     LD   B,A
1406     EX  DE,HL
1407     CALL PON
1408     LD   B,1
1409     JP   P_L
1410 PON   LD   A,32
1411     LD   (HL),A
1412     INC  HL

```

PROGRAMAS

```

1413      DJNZ PON
1414      RET
1415 IN_LI LD   HL, 51320
1416      LD   B, 64
1417 INLI  LD   A, (HL)
1418      INC  HL
1419      CP   32
1420      JR   NZ, F_INLI
1421      DJNZ INLI
1422      JR   FIN_SP
1423 F_INLI RET
1424 FIN_SP LD   A, (CX)
1425      PUSH AF
1426      XOR  A
1427      LD   (CX), A
1428      CALL HL_COR
1429      LD   DE, 51384
1430      EX  DE, HL
1431      AND  A
1432      SBC HL, DE
1433      LD   B, H
1434      LD   C, L
1435      LD   DE, 51383
1436      LD   HL, 51319
1437      LDDR
1438      LD   B, 64
1439      LD   A, 32
1440 PON_SP LD   (DE), A
1441      INC  DE
1442      DJNZ PON_SP
1443      POP  AF
1444      LD   (CX), A
1445      CALL SCREEN
1446      JP   BEEP
1447 LI_DE  LD   A, (CX)
1448      LD   D, A
1449      LD   A, (CY)
1450      LD   E, A
1451      PUSH DE
1452      INC  A
1453      LD   (CY), A
1454      XOR  A
1455      LD   (CX), A
1456      CALL HL_COR
1457      DEC  HL
1458      LD   A, (HL)
1459      CP   32
1460      JR   Z, SI_DE
1461 FIN_DE POP  DE
1462      LD   A, D
1463      LD   (CX), A
1464      LD   A, E
1465      LD   (CY), A
1466      JP   BEEP
1467 SI_DE  PUSH HL
1468      POP  DE
1469      DEC  HL
1470      LD   BC, 63
1471      LDDR
1472      LD   A, 32
1473      LD   (DE), A
1474      LD   A, (CY)
1475      DEC  A
1476      LD   (CY), A
1477      CALL LOP_
1478      CALL CL_LIN
1479      CALL P_L_C
1480      JR   FIN_DE
1481 LI_IZ  LD   A, (CX)

```

```

1482      PUSH AF
1483      XOR  A
1484      LD   (CX), A
1485      CALL HL_COR
1486      LD   A, (HL)
1487      CP   32
1488      JR   Z, SI_IZ
1489 FIN_IZ POP  AF
1490      LD   (CX), A
1491      JP   BEEP
1492 SI_IZ  PUSH HL
1493      POP  DE
1494      INC  HL
1495      LD   BC, 63
1496      LDIR
1497      LD   A, 32
1498      LD   (DE), A
1499      CALL LOP_
1500      CALL CL_LIN
1501      CALL P_L_C
1502      JR   FIN_IZ
1503 EXT_M  LD   A, (23617)
1504      AND  A
1505      JR   Z, N_EXT
1506      XOR  A
1507      LD   (23617), A
1508      CALL UNO
1509      LD   A, 22
1510      RST  16
1511      LD   A, 1
1512      RST  16
1513      LD   A, 22
1514      RST  16
1515      LD   A, 78
1516      RST  16
1517      LD   A, 79
1518      JR   F_EXT
1519 N_EXT  LD   A, 1
1520      LD   (23617), A
1521      CALL UNO
1522      LD   A, 22
1523      RST  16
1524      LD   A, 1
1525      RST  16
1526      LD   A, 22
1527      RST  16
1528      LD   A, 83
1529      RST  16
1530      LD   A, 73
1531 F_EXT  RST  16
1532      CALL DOS
1533      JP   BEEP
1534 UNO    LD   A, 1
1535      CALL 5633
1536      JP   OVER_0
1537 DOS    LD   A, 2
1538      CALL 5633
1539      JP   OVER_1
1540 CU_PR  LD   B, 22
1541      XOR  A
1542      LD   (CY), A
1543      LD   (CX), A
1544      LD   (C_Y), A
1545      LD   DE, 0
1546      PUSH DE
1547      PUSH AF
1548      JP   P_L
1549 CU_FI  LD   B, 22
1550      LD   A, 234

```

```

1551 LD (CY), A
1552 LD E, 255
1553 XOR A
1554 LD D, A
1555 LD (C_Y), A
1556 LD (CX), A
1557 PUSH DE
1558 LD A, 21
1559 PUSH AF
1560 JP P_L
1561 DE_LI CALL HL_COR
1562 LD D, 0
1563 LD A, (CX)
1564 LD E, A
1565 AND A
1566 SBC HL, DE
1567 PUSH HL
1568 LD DE, 64
1569 ADD HL, DE
1570 PUSH HL
1571 LD DE, 35000
1572 AND A
1573 SBC HL, DE
1574 LD DE, 16384
1575 EX DE, HL
1576 AND A
1577 SBC HL, DE
1578 LD B, H
1579 LD C, L
1580 POP HL
1581 POP DE
1582 LD A, B
1583 OR C
1584 JR Z, L_255
1585 LDIR
1586 L_255 LD HL, 35000
1587 LD DE, 16320
1588 ADD HL, DE
1589 PUSH HL
1590 POP DE
1591 INC DE
1592 LD BC, 63
1593 LD (HL), 32
1594 LDIR
1595 CALL SCREEN
1596 JP BEEP
1597 SCREEN LD A, (CX)
1598 LD D, A
1599 LD A, (CY)
1600 LD E, A
1601 PUSH DE
1602 LD A, (C_Y)
1603 PUSH AF
1604 LD B, A
1605 LD A, 21
1606 SUB B
1607 LD B, A
1608 INC B
1609 P_L PUSH BC
1610 CALL LOP_
1611 CALL CL_LIN
1612 CALL P_L_C
1613 LD A, (CY)
1614 INC A
1615 LD (CY), A
1616 LD A, (C_Y)
1617 INC A
1618 LD (C_Y), A
1619 POP BC

```

```

1620 DJNZ P_L
1621 POP POP AF
1622 LD (C_Y), A
1623 POP DE
1624 LD A, D
1625 LD (CX), A
1626 LD A, E
1627 LD (CY), A
1628 RET
1629 LOP_ LD A, (C_Y)
1630 CP 16
1631 JR NC, MAYOR
1632 CP 8
1633 JR NC, MASS
1634 LD HL, #4000
1635 LOP_0 AND A
1636 RET Z
1637 LD B, A
1638 LD DE, #20
1639 LOP_1 ADD HL, DE
1640 DJNZ LOP_1
1641 RET
1642 MAYOR LD HL, #5000
1643 SUB 16
1644 JR LOP_0
C1645 MASS LD HL, #4800
1646 SUB 8
1647 JR LOP_0
1648 SCR_D LD B, 5
1649 LD HL, #579F
1650 LD DE, #57BF
1651 CALL SCR
1652 LD B, 1
1653 LD HL, #4FFF
1654 LD DE, #571F
1655 CALL SCR
1656 LD B, 7
1657 LD HL, #4FDF
1658 LD DE, #4FFF
1659 CALL SCR
1660 LD B, 1
1661 LD HL, #47FF
1662 LD DE, #4F1F
1663 CALL SCR
1664 LD B, 7
1665 LD HL, #47DF
1666 LD DE, #47FF
1667 CALL SCR
1668 LD HL, #4000
1669 CALL CL_LIN
1670 CALL P_L_C
1671 RET
1672 SCR PUSH BC
1673 LD B, 8
1674 PUSH HL
1675 PUSH DE
1676 SCR_0 PUSH BC
1677 PUSH HL
1678 PUSH DE
1679 LD BC, #20
1680 LDDR
1681 POP DE
1682 POP HL
1683 DEC H
1684 DEC D
1685 POP BC
1686 DJNZ SCR_0
1687 POP DE
1688 EX DE, HL

```

PROGRAMAS

```

1689 LD DE, #20
1690 AND A
1691 SBC HL, DE
1692 EX DE, HL
1693 POP HL
1694 PUSH DE
1695 LD DE, #20
1696 AND A
1697 SBC HL, DE
1698 POP DE
1699 POP BC
1700 DJNZ SCR
1701 RET
1702 SCR_U LD B, 7
1703 LD HL, #4020
1704 LD DE, #4000
1705 CALL SCR^
1706 LD B, 1
1707 LD HL, #4800
1708 LD DE, #40E0
1709 CALL SCR^
1710 LD B, 7
1711 LD HL, #4820
1712 LD DE, #4800
1713 CALL SCR^
1714 LD B, 1
1715 LD HL, #5000
1716 LD DE, #48E0
1717 CALL SCR^
1718 LD B, 5
1719 LD HL, #5020
1720 LD DE, #5000
1721 CALL SCR^
1722 LD HL, #50A0
1723 CALL CL_LIN
1724 CALL P_L_C
1725 RET
1726 P_L_C LD HL, 35000
1727 LD A, (CY)
1728 AND A
1729 JR Z, LINEA
1730 LD B, A
1731 LD DE, 64
1732 X64 ADD HL, DE
1733 DJNZ X64
1734 LINEA LD B, 64
1735 LD A, (CX)
1736 PUSH AF
1737 XOR A
1738 LD (CX), A
1739 PRT_L PUSH BC
1740 PUSH HL
1741 CALL AT
1742 POP HL
1743 LD A, (HL)
1744 LD (LAST_K), A
1745 LD A, (CX)
1746 SRL A
1747 PUSH HL
1748 JR C, IMPAR
1749 CALL IZQ
1750 JR MASAU
1751 IMPAR LD A, (LAST_K)
1752 RST 16
1753 MASAU POP HL
1754 LD A, (CX)
1755 INC A
1756 LD (CX), A
1757 POP BC

```

```

1758 INC HL
1759 DJNZ PRT_L
1760 POP AF
1761 LD (CX), A
1762 RET
1763 CL_LIN LD B, 8
1764 CL_1 PUSH BC
1765 LD BC, 31
1766 PUSH HL
1767 PUSH HL
1768 POP DE
1769 INC DE
1770 LD (HL), 0
1771 LDIR
1772 POP HL
1773 INC H
C1774 POP BC
1775 DJNZ CL_1
1776 RET
1777 SCR^ PUSH BC
1778 LD B, 8
1779 PUSH HL
1780 PUSH DE
1781 SCR^_0 PUSH BC
1782 PUSH HL
1783 PUSH DE
1784 LD BC, 32
1785 LDIR
1786 POP DE
1787 POP HL
1788 INC H
1789 INC D
1790 POP BC
1791 DJNZ SCR^_0
1792 POP DE
1793 EX DE, HL
1794 LD DE, 32
1795 ADD HL, DE
1796 EX DE, HL
1797 POP HL
1798 PUSH DE
1799 LD DE, 32
1800 ADD HL, DE
1801 POP DE
1802 POP BC
1803 DJNZ SCR^
1804 RET
1805 COMP_1 LD A, (CX)
1806 CP 63
1807 RET NZ
1808 COMP_F LD A, (CY)
1809 CP 255
1810 RET NZ
1811 POP AF
1812 RET
1813 AT LD A, 22
1814 RST 16
1815 LD A, (C_Y)
1816 RST 16
1817 LD A, (CX)
1818 SRL A
1819 RST 16
1820 RET
1821 OVER_1 LD A, 21
1822 RST 16
1823 LD A, 1
1824 RST 16
1825 RET
1826 OVER_0 LD A, 21

```

```

1827      RST 16
1828      XOR A
1829      RST 16
1830      RET
1831 BEEP  LD DE,123
1832      LD HL,388
1833      JP 949
1834 ENTER CALL COMP_F
1835      CALL NEW_L
1836      CALL BEEP
1837      RET
1838 DELETE CALL AT
1839      CALL CL_CHR
1840      CALL HL_COR
1841      LD (HL),32
1842      CALL BEEP
1843      CALL COMP
1844      LD A,(CX)
1845      AND A
1846      JR Z,N_COL
1847      DEC A
1848      LD (CX),A
1849      RET
1850 N_COL LD A,63
1851      LD (CX),A
1852 IF_SCR LD A,(CY)
1853      DEC A
1854      LD (CY),A
1855      LD A,(C_Y)
1856      AND A
1857      PUSH AF
1858      CALL Z,SCR_D
1859      POP AF
1860      RET Z
1861      DEC A
1862      LD (C_Y),A
1863      RET
1864 CL_CHR LD A,(CX)
1865      SRL A
1866      PUSH AF
1867      CALL HL_COR
1868      POP AF
1869      LD A,(HL)
1870      JR C,B_DER
1871      LD (LAST_K),A
1872      JP IZQ
1873 B_DER RST 16
1874      RET
1875 COMP  LD A,(CY)
1876      AND A
1877      RET NZ
1878      LD A,(CX)
1879      AND A
1880      RET NZ
1881      POP AF
1882      RET
1883 CP_L  CALL BEEP
1884      LD A,(23658)
1885      BIT 3,A
1886      JR NZ,NO_CP
1887      LD A,8
1888      LD (23658),A
1889      CALL UNO
1890      LD A,22
1891      RST 16
1892      LD A,1
1893      RST 16
1894      LD A,8
1895      RST 16

```

```

1896      LD A,83
1897      RST 16
1898      LD A,73
1899      RST 16
1900      LD A,32
1901      RST 16
1902      JP DOS
C1903 NO_CP XOR A
1904      LD (23658),A
1905      CALL UNO
1906      LD A,22
1907      RST 16
1908      LD A,1
1909      RST 16
1910      LD A,8
1911      RST 16
1912      LD A,78
1913      RST 16
1914      LD A,79
1915      RST 16
1916      LD A,32
1917      RST 16
1918      JP DOS
1919 P_NCOL LD A,(COORDS)
1920      LD B,0
1921 REST_  INC B
1922      SUB 100
1923      JR NC,REST_
1924      DEC B
1925      LD A,B
1926      AND A
1927      JR Z,RES_1
1928      PUSH AF
1929      LD A,(COORDS)
1930 REST_1 SUB 100
1931      DJNZ REST_1
1932      LD (COORDS),A
1933      POP AF
1934 RES_1  ADD A,48
1935      RST 16
1936 REST_0 LD A,(COORDS)
1937      LD B,0
1938 REST  INC B
1939      SUB 10
1940      JR NC,REST
1941      DEC B
1942      LD A,B
1943      PUSH BC
1944      ADD A,48
1945      RST 16
1946      POP BC
1947      XOR A
1948 X10   ADD A,10
1949      DJNZ X10
1950      LD B,A
1951      LD A,(COORDS)
1952      SUB B
1953      ADD A,48
1954 P_DIG RST 16
1955      JP DOS
1956 *D+
1957 CUR_L CALL BEEP
1958      CALL COMP
1959      LD A,(CX)
1960      AND A
1961      JR Z,N_LEFT
1962      DEC A
1963      LD (CX),A
1964      RET

```

PROGRAMAS

1965	N_LEFT	LD	A, 63
1966		LD	(CX), A
1967		JP	IF_SCR
1968	CUR_R	CALL	COMP_1
1969		LD	A, (CX)
1970		CP	63
1971		JR	Z, N_RIGH
1972		INC	A
1973		LD	(CX), A
1974		JP	BEEP
1975	N_RIGH	XOR	A
1976		LD	(CX), A
1977	CUR_D	CALL	COMP_F
1978		CALL	BEEP
1979		LD	A, (CY)
1980		INC	A
1981		LD	(CY), A
1982		LD	A, (C_Y)
1983		CP	21
1984		PUSH	AF
1985		CALL	Z, SCR_U
1986		POP	AF
1987		RET	Z
1988		INC	A
1989		LD	(C_Y), A
1990		RET	
1991	CUR_U	LD	A, (CY)
1992		AND	A
1993		JP	Z, BEEP
1994		CALL	BEEP
1995		JP	IF_SCR
1996	CURSOR	CALL	AT
1997		LD	A, (CX)
1998		SRL	A
1999		JR	C, CU_RI
2000		LD	A, 138
2001		RST	16
2002		RET	
2003	CU_RI	LD	A, 133
2004		RST	16
2005		RET	
2006	MARG_D	CALL	BEEP
2007		LD	A, (CX)
2008		LD	HL, MI
2009		CP	(HL)
2010		RET	C
2011		LD	(MD), A
2012		LD	(COORDS), A
2013	SET_MD	LD	A, 1
2014		CALL	5633
2015		LD	A, 22
2016		RST	16
2017		LD	A, 1
2018		RST	16
2019		LD	A, 17
2020		RST	16
2021		LD	A, (MD)
2022		LD	(COORDS), A
2023		CALL	P_NCOL
2024		RET	
2025	MARG_I	CALL	BEEP
2026		LD	A, (CX)
2027		LD	HL, MD
2028		CP	(HL)
2029		RET	NC
2030		LD	(MI), A
2031	SET_MI	CALL	UNO
2032		LD	A, 22
2033		RST	16
2034		LD	A, 1
2035		RST	16
2036		LD	A, 12
2037		RST	16
2038		LD	A, (MI)
2039		LD	(COORDS), A
2040		JP	P_NCOL
2041	CINTA	CALL	S^
2042		LD	HL, CASETE
2043		CALL	RS_INS
2044	OTR	CALL	KEY
2045		CP	76
2046		JR	Z, LOAD
2047		CP	108
2048		JR	Z, LOAD
2049		CP	83
2050		JP	Z, SAVE
2051		CP	115
2052		JP	Z, SAVE
2053		JR	OTR
2054	PONES	DEFB	22, 21, 1
2055		DEFM	'PON EL CASSET
			E EN MARCHA.'
2056		DEFB	0, 0
2057	PONE	DEFB	22, 20, 1
2058		DEFM	'PON RECORD EN
			EL CASSETE.'
2059		DEFB	13
2060		DEFM	'PULSA UNA TEC LA.'
2061		DEFB	0, 0
2062	CASETE	DEFB	22, 8, 10
2063		DEFM	'S= SALVAR.'
2064		DEFB	22, 12, 10
2065		DEFM	'L= CARGAR.'
2066		DEFB	0, 0
2067	LOAD	CALL	3435
2068		LD	HL, PONES
2069		CALL	RS_INS
2070		LD	IX, 35000
2071		LD	DE, 16384
2072		LD	A, 128
2073		SCF	
2074		CALL	1366
2075		JR	NC, LOAD
2076		JR	S^^
2077	SAVE	CALL	3435
2078		LD	HL, PONE
2079		CALL	RS_INS
2080	KEY_S	CALL	KEY
2081		AND	A
2082		JR	Z, KEY_S
2083		LD	IX, 35000
2084		LD	DE, 16384
2085		LD	A, 128
2086		SCF	
2087		CALL	1218
2088		JR	S^^
2089	S^	LD	DE, 55000
2090		LD	HL, 16384
2091		LD	BC, 6912
2092		LDIR	
2093		CALL	3435
2094		CALL	DOS
2095		LD	A, 60
2096		LD	(23607), A
2097		RET	
2098	INSTR	CALL	S^
2099		LD	HL, INTR
2100		CALL	RS_INS
2101	END_	CALL	KEY

```

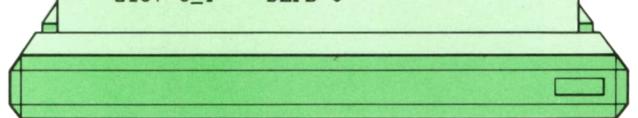
2102 AND A
2103 JR Z,END_
2104 JP S^^
2105 RS_INS LD A,(HL)
2106 AND A
2107 JR Z,END_I
2108 RST RST 16
2109 INC HL
2110 JR RS_INS
2111 END_I INC HL
2112 LD A,(HL)
2113 DEC HL
2114 AND A
2115 LD A,0
2116 JR NZ,RST
2117 RET
2118 S^^ LD A,128
2119 LD (23607),A
2120 LD HL,55000
2121 LD DE,16384
2122 LD BC,6912
2123 LDIR
2124 JP DOS
2125 INTR DEFB 20,1
2126 DEFM 'MODO NORMAL:'
2127 DEFB 20,0,13
2128 DEFM '-CAPS SHIFT+1
=INSTRUCCIONES.'
2129 DEFB 13
2130 DEFM '-CAPS SHIFT+2
=MAYUSC./MINUSC.'
2131 DEFB 13
2132 DEFM '- " +3
=MARGEN DERECHO.'
2133 DEFB 13
2134 DEFM '- " +4
=MARGEN IZQUIER.'
2135 DEFB 13
2136 DEFM '- " +9
=SAVE/LOAD'
2137 DEFB 13
2138 DEFM '-S.SHIFT+3=N'
2139 DEFB 8
2140 DEFB 126
2141 DEFB 13,13,20,1
2142 DEFM 'MODO EXTENDID
O:'
2143 DEFB 20,0,13
2144 DEFM '-0=BORRA LINE
A EN CURSO.'
2145 DEFB 13
2146 DEFM '-1=LINEA A LA
IZQUIERDA.'
2147 DEFB 13

```

```

C2148 DEFM '-2=LINEA A LA
DERECHA.'
2149 DEFB 13
2150 DEFM '-3=INSERTA LI
NEA.'
2151 DEFB 13
2152 DEFM '-4=INSERTA CA
RACTER.'
2153 DEFB 13
2154 DEFM '-5=CENTRA LIN
EA.'
2155 DEFB 13
2156 DEFM '-6=CURSOR AL
FINAL.'
2157 DEFB 13
2158 DEFM '-7=CURSOR AL
PRINCIPIO.'
2159 DEFB 13
2160 DEFM '-8=BORRA CARA
CTER.'
2161 DEFB 13
2162 DEFM '-9=PASA POR I
MPRESORA.'
2163 DEFB 13
2164 DEFM '-A,S,D,F,G=VO
CALES ACENTUADAS.'
2165 DEFB 13,13,18,1
2166 DEFM 'PULSA UNA TEC
LA PARA VOLVER.'
2167 DEFB 18,0
2168 DEFM ' '
2169 DEFB 0,0
2170 LOWER DEFM 'COLUMNA '
2171 DEFM 'LINEA '
2172 DEFM 'CAPS L. '
2173 DEFM 'MARG.'
2174 DEFM 'IZQ. '
2175 DEFM 'MARG.'
2176 DEFM 'DER. '
2177 DEFM 'M.EXTEND. '
2178 DEFM 'EDIT=INST. '
2179 CX DEFB 0
2180 CY DEFB 0
2181 MI DEFB 0
2182 MD DEFB 63
2183 D_SET DEFW 32768
2184 LAST_K DEFB 0
2185 COORDS DEFB 0
2186 *L+
2187 C_Y DEFB 0

```



Para terminar, aparece el listado fuente del programa en código máquina

para aquellos que deseen ver su funcionamiento.

```

7000 DATA 62,128,50,55,92,62,8,50,106,92,205,230,122,205,12,1479
7002 DATA 123,62,1,50,65,92,205,17,119,62,1,205,1,22,205,1230
7004 DATA 113,121,33,13,126,58,79,126,245,175,50,79,126,205,229,1778
7006 DATA 120,241,50,79,126,205,83,119,58,80,126,50,78,126,205,1746
7008 DATA 233,121,205,140,115,6,65,197,6,255,54,32,35,16,251,1731
7010 DATA 193,16,245,24,43,33,184,136,58,78,126,22,0,95,213,1466
7012 DATA 58,79,126,167,40,7,17,64,0,71,25,16,253,209,25,1157
7014 DATA 201,205,226,115,167,245,196,238,115,205,195,122,205,195,122,2752
7016 DATA 241,40,239,58,78,126,50,85,126,205,75,119,62,22,215,1741
7018 DATA 62,1,215,175,215,205,44,122,58,79,126,50,85,126,205,1768

```

7020 DATA 75, 119, 62, 22, 215, 62, 1, 215, 62, 4, 215, 205, 44, 122, 24, 1447
 7022 DATA 196, 175, 50, 8, 92, 255, 58, 8, 92, 50, 84, 126, 201, 254, 128, 1777
 7024 DATA 208, 254, 13, 202, 135, 121, 254, 12, 202, 145, 121, 254, 24, 202, 145, 2292
 7026 DATA 121, 254, 6, 202, 233, 121, 254, 26, 202, 233, 121, 254, 15, 202, 33, 2277
 7028 DATA 123, 254, 1, 202, 33, 123, 254, 8, 202, 107, 122, 254, 9, 202, 132, 2026
 7030 DATA 122, 254, 10, 202, 153, 122, 254, 11, 202, 182, 122, 254, 7, 202, 249, 2346
 7032 DATA 123, 254, 25, 202, 249, 123, 254, 5, 202, 213, 122, 254, 30, 200, 254, 2510
 7034 DATA 29, 200, 254, 31, 200, 254, 28, 202, 213, 122, 254, 4, 202, 254, 122, 2369
 7036 DATA 254, 27, 202, 254, 122, 254, 14, 202, 17, 119, 254, 16, 202, 135, 119, 2191
 7038 DATA 254, 17, 202, 228, 118, 254, 18, 202, 161, 118, 254, 22, 202, 111, 119, 2280
 7040 DATA 254, 23, 202, 91, 119, 254, 19, 202, 99, 118, 254, 20, 202, 157, 117, 2131
 7042 DATA 254, 21, 202, 231, 117, 254, 2, 202, 124, 117, 254, 3, 202, 29, 117, 2129
 7044 DATA 205, 126, 121, 205, 113, 121, 205, 99, 121, 58, 84, 126, 245, 205, 200, 2234
 7046 DATA 121, 241, 50, 84, 126, 245, 205, 140, 115, 241, 119, 205, 99, 121, 58, 2170
 7048 DATA 78, 126, 203, 63, 56, 5, 212, 181, 116, 24, 58, 58, 84, 126, 215, 1605
 7050 DATA 24, 52, 42, 82, 126, 22, 0, 58, 84, 126, 95, 6, 8, 25, 16, 766
 7052 DATA 253, 84, 93, 6, 8, 126, 203, 39, 203, 39, 203, 39, 203, 39, 119, 1657
 7054 DATA 35, 16, 243, 58, 84, 126, 215, 235, 6, 8, 126, 203, 63, 203, 63, 1684
 7056 DATA 203, 63, 203, 63, 119, 35, 16, 243, 201, 205, 85, 121, 58, 78, 126, 1819
 7058 DATA 60, 33, 81, 126, 190, 56, 2, 32, 8, 254, 64, 40, 4, 50, 78, 1078
 7060 DATA 126, 201, 58, 80, 126, 50, 78, 126, 58, 79, 126, 60, 50, 79, 126, 1423
 7062 DATA 58, 86, 126, 254, 21, 245, 204, 148, 120, 241, 200, 60, 50, 86, 126, 2025
 7064 DATA 201, 205, 226, 123, 62, 128, 50, 55, 92, 58, 78, 126, 87, 58, 79, 1628
 7066 DATA 126, 95, 213, 58, 86, 126, 245, 175, 50, 86, 126, 50, 78, 126, 50, 1690
 7068 DATA 79, 126, 6, 11, 197, 205, 199, 119, 245, 197, 213, 229, 205, 172, 14, 2217
 7070 DATA 225, 209, 193, 241, 58, 79, 126, 198, 22, 50, 79, 126, 193, 16, 230, 2045
 7072 DATA 62, 8, 50, 86, 126, 205, 199, 119, 6, 112, 33, 0, 72, 243, 205, 1526
 7074 DATA 178, 14, 205, 28, 124, 241, 50, 86, 126, 209, 122, 50, 78, 126, 123, 1760
 7076 DATA 50, 79, 126, 195, 126, 121, 205, 140, 115, 229, 209, 35, 58, 78, 126, 1892
 7078 DATA 71, 62, 64, 144, 6, 0, 79, 237, 176, 62, 32, 18, 205, 3, 120, 1279
 7080 DATA 205, 27, 121, 205, 213, 120, 195, 128, 121, 58, 79, 126, 245, 50, 79, 1970
 7082 DATA 126, 58, 78, 126, 245, 62, 64, 50, 78, 126, 205, 140, 115, 241, 50, 1764
 7084 DATA 78, 126, 241, 50, 79, 126, 43, 126, 254, 32, 192, 235, 213, 225, 43, 2063
 7086 DATA 58, 78, 126, 71, 62, 63, 144, 79, 6, 0, 237, 184, 205, 126, 121, 1560
 7088 DATA 205, 140, 115, 54, 32, 58, 78, 126, 87, 58, 79, 126, 95, 213, 58, 1524
 7090 DATA 86, 126, 245, 6, 1, 195, 218, 119, 58, 78, 126, 87, 58, 79, 126, 1608
 7092 DATA 95, 213, 58, 86, 126, 245, 175, 50, 78, 126, 205, 140, 115, 6, 0, 1718
 7094 DATA 126, 35, 4, 254, 32, 40, 249, 197, 62, 63, 50, 78, 126, 205, 140, 1661
 7096 DATA 115, 14, 0, 126, 43, 12, 254, 32, 40, 249, 209, 66, 5, 13, 120, 1298
 7098 DATA 129, 87, 62, 64, 146, 245, 175, 50, 78, 126, 197, 205, 140, 115, 193, 2012
 7100 DATA 22, 0, 88, 25, 17, 54, 91, 241, 197, 6, 0, 79, 197, 237, 176, 1430
 7102 DATA 209, 193, 213, 120, 129, 203, 63, 245, 205, 140, 115, 241, 71, 245, 205, 2597
 7104 DATA 92, 118, 235, 33, 54, 91, 241, 193, 237, 176, 71, 235, 205, 92, 118, 2191
 7106 DATA 6, 1, 195, 218, 119, 62, 32, 119, 35, 16, 250, 201, 33, 120, 200, 1607
 7108 DATA 6, 64, 126, 35, 254, 32, 32, 4, 16, 248, 24, 1, 201, 58, 78, 1179
 7110 DATA 126, 245, 175, 50, 78, 126, 205, 140, 115, 17, 184, 200, 235, 167, 237, 2300
 7112 DATA 82, 68, 77, 17, 183, 200, 33, 119, 200, 237, 184, 6, 64, 62, 32, 1564
 7114 DATA 18, 19, 16, 252, 241, 50, 78, 126, 205, 199, 119, 195, 126, 121, 58, 1823
 7116 DATA 78, 126, 87, 58, 79, 126, 95, 213, 60, 50, 79, 126, 175, 50, 78, 1480
 7118 DATA 126, 205, 140, 115, 43, 126, 254, 32, 40, 12, 209, 122, 50, 78, 126, 1678
 7120 DATA 123, 50, 79, 126, 195, 126, 121, 229, 209, 43, 1, 63, 0, 237, 184, 1786
 7122 DATA 62, 32, 18, 58, 79, 126, 61, 50, 79, 126, 205, 3, 120, 205, 27, 1251
 7124 DATA 121, 205, 213, 120, 24, 215, 58, 78, 126, 245, 175, 50, 78, 126, 205, 2039
 7126 DATA 140, 115, 126, 254, 32, 40, 7, 241, 50, 78, 126, 195, 126, 121, 229, 1880
 7128 DATA 209, 35, 1, 63, 0, 237, 176, 62, 32, 18, 205, 3, 120, 205, 27, 1393
 7130 DATA 121, 205, 213, 120, 24, 227, 58, 65, 92, 167, 40, 23, 175, 50, 65, 1645
 7132 DATA 92, 205, 75, 119, 62, 22, 215, 62, 1, 215, 62, 22, 215, 62, 78, 1507
 7134 DATA 215, 62, 79, 24, 22, 62, 1, 50, 65, 92, 205, 75, 119, 62, 22, 1155
 7136 DATA 215, 62, 1, 215, 62, 22, 215, 62, 83, 215, 62, 73, 215, 205, 83, 1790
 7138 DATA 119, 195, 126, 121, 62, 1, 205, 1, 22, 195, 120, 121, 62, 2, 205, 1557
 7140 DATA 1, 22, 195, 113, 121, 6, 22, 175, 50, 79, 126, 50, 78, 126, 50, 1214
 7142 DATA 86, 126, 17, 0, 0, 213, 245, 195, 218, 119, 6, 22, 62, 234, 50, 1593
 7144 DATA 79, 126, 30, 255, 175, 87, 50, 86, 126, 50, 78, 126, 213, 62, 21, 1564
 7146 DATA 245, 195, 218, 119, 205, 140, 115, 22, 0, 58, 78, 126, 95, 167, 237, 2020
 7148 DATA 82, 229, 17, 64, 0, 25, 229, 17, 184, 136, 167, 237, 82, 17, 0, 1486
 7150 DATA 64, 235, 167, 237, 82, 68, 77, 225, 209, 120, 177, 40, 2, 237, 176, 2116
 7152 DATA 33, 184, 136, 17, 192, 63, 25, 229, 209, 19, 1, 63, 0, 54, 32, 1257
 7154 DATA 237, 176, 205, 199, 119, 195, 126, 121, 58, 78, 126, 87, 58, 79, 126, 1990

7156 DATA 95, 213, 58, 86, 126, 245, 71, 62, 21, 144, 71, 4, 197, 205, 3, 1601
 7158 DATA 120, 205, 27, 121, 205, 213, 120, 58, 79, 126, 60, 50, 79, 126, 58, 1647
 7160 DATA 86, 126, 60, 50, 86, 126, 193, 16, 229, 241, 50, 86, 126, 209, 122, 1806
 7162 DATA 50, 78, 126, 123, 50, 79, 126, 201, 58, 86, 126, 254, 16, 48, 17, 1438
 7164 DATA 254, 8, 48, 20, 33, 0, 64, 167, 200, 71, 17, 32, 0, 25, 16, 955
 7166 DATA 253, 201, 33, 0, 80, 214, 16, 24, 239, 33, 0, 72, 214, 8, 24, 1411
 7168 DATA 232, 6, 5, 33, 159, 87, 17, 191, 87, 205, 106, 120, 6, 1, 33, 1288
 7170 DATA 255, 79, 17, 31, 87, 205, 106, 120, 6, 7, 33, 223, 79, 17, 255, 1520
 7172 DATA 79, 205, 106, 120, 6, 1, 33, 255, 71, 17, 31, 79, 205, 106, 120, 1434
 7174 DATA 6, 7, 33, 223, 71, 17, 255, 71, 205, 106, 120, 33, 0, 64, 205, 1416
 7176 DATA 27, 121, 205, 213, 120, 201, 197, 6, 8, 229, 213, 197, 229, 213, 1, 2180
 7178 DATA 32, 0, 237, 184, 209, 225, 37, 21, 193, 16, 241, 209, 235, 17, 32, 1888
 7180 DATA 0, 167, 237, 82, 235, 225, 213, 17, 32, 0, 167, 237, 82, 209, 193, 2096
 7182 DATA 16, 215, 201, 6, 7, 33, 32, 64, 17, 0, 64, 205, 47, 121, 6, 1034
 7184 DATA 1, 33, 0, 72, 17, 224, 64, 205, 47, 121, 6, 7, 33, 32, 72, 934
 7186 DATA 17, 0, 72, 205, 47, 121, 6, 1, 33, 0, 80, 17, 224, 72, 205, 1100
 7188 DATA 47, 121, 6, 5, 33, 32, 80, 17, 0, 80, 205, 47, 121, 33, 160, 987
 7190 DATA 80, 205, 27, 121, 205, 213, 120, 201, 33, 184, 136, 58, 79, 126, 167, 1955
 7192 DATA 40, 7, 71, 17, 64, 0, 25, 16, 253, 6, 64, 58, 78, 126, 245, 1070
 7194 DATA 175, 50, 78, 126, 197, 229, 205, 99, 121, 225, 126, 50, 84, 126, 58, 1949
 7196 DATA 78, 126, 203, 63, 229, 56, 5, 205, 181, 116, 24, 4, 58, 84, 126, 1558
 7198 DATA 215, 225, 58, 78, 126, 60, 50, 78, 126, 193, 35, 16, 217, 241, 50, 1768
 7200 DATA 78, 126, 201, 6, 8, 197, 1, 31, 0, 229, 229, 209, 19, 54, 0, 1388
 7202 DATA 237, 176, 225, 36, 193, 16, 239, 201, 197, 6, 8, 229, 213, 197, 229, 2402
 7204 DATA 213, 1, 32, 0, 237, 176, 209, 225, 36, 20, 193, 16, 241, 209, 235, 2043
 7206 DATA 17, 32, 0, 25, 235, 225, 213, 17, 32, 0, 25, 209, 193, 16, 219, 1458
 7208 DATA 201, 58, 78, 126, 254, 63, 192, 58, 79, 126, 254, 255, 192, 241, 201, 2378
 7210 DATA 62, 22, 215, 58, 86, 126, 215, 58, 78, 126, 203, 63, 215, 201, 62, 1790
 7212 DATA 21, 215, 62, 1, 215, 201, 62, 21, 215, 175, 215, 201, 17, 123, 0, 1744
 7214 DATA 33, 132, 1, 195, 181, 3, 205, 91, 121, 205, 0, 117, 205, 126, 121, 1736
 7216 DATA 201, 205, 99, 121, 205, 200, 121, 205, 140, 115, 54, 32, 205, 126, 121, 2150
 7218 DATA 205, 221, 121, 58, 78, 126, 167, 40, 5, 61, 50, 78, 126, 201, 62, 1599
 7220 DATA 63, 50, 78, 126, 58, 79, 126, 61, 50, 79, 126, 58, 86, 126, 167, 1333
 7222 DATA 245, 204, 41, 120, 241, 200, 61, 50, 86, 126, 201, 58, 78, 126, 203, 2040
 7224 DATA 63, 245, 205, 140, 115, 241, 126, 58, 6, 50, 84, 126, 195, 181, 116, 1949
 7226 DATA 215, 201, 58, 79, 126, 167, 192, 58, 78, 126, 167, 192, 241, 201, 205, 2306
 7228 DATA 126, 121, 58, 106, 92, 203, 95, 32, 29, 62, 8, 50, 106, 92, 205, 1385
 7230 DATA 75, 119, 62, 22, 215, 62, 1, 215, 62, 8, 215, 62, 83, 215, 62, 1478
 7232 DATA 73, 215, 62, 32, 215, 195, 83, 119, 175, 50, 106, 92, 205, 75, 119, 1816
 7234 DATA 62, 22, 215, 62, 1, 215, 62, 8, 215, 62, 78, 215, 62, 79, 215, 1573
 7236 DATA 62, 32, 215, 195, 83, 119, 58, 85, 126, 6, 0, 4, 214, 100, 48, 1347
 7238 DATA 251, 5, 120, 167, 40, 12, 245, 58, 85, 126, 214, 100, 16, 252, 50, 1741
 7240 DATA 85, 126, 241, 198, 48, 215, 58, 85, 126, 6, 0, 4, 214, 10, 48, 1464
 7242 DATA 251, 5, 120, 197, 198, 48, 215, 193, 175, 198, 10, 16, 252, 71, 58, 2007
 7244 DATA 85, 126, 144, 198, 48, 215, 195, 83, 119, 205, 126, 121, 205, 221, 121, 2212
 7246 DATA 58, 78, 126, 167, 40, 5, 61, 50, 78, 126, 201, 62, 63, 50, 78, 1243
 7248 DATA 126, 195, 178, 121, 205, 85, 121, 58, 78, 126, 254, 63, 40, 7, 60, 1717
 7250 DATA 50, 78, 126, 195, 126, 121, 175, 50, 78, 126, 205, 91, 121, 205, 126, 1873
 7252 DATA 121, 58, 79, 126, 60, 50, 79, 126, 58, 86, 126, 254, 21, 245, 204, 1693
 7254 DATA 148, 120, 241, 200, 60, 50, 86, 126, 201, 58, 79, 126, 167, 202, 126, 1990
 7256 DATA 121, 205, 126, 121, 195, 178, 121, 205, 99, 121, 58, 78, 126, 203, 63, 2020
 7258 DATA 56, 4, 62, 138, 215, 201, 62, 133, 215, 201, 205, 126, 121, 58, 78, 1875
 7260 DATA 126, 33, 80, 126, 190, 216, 50, 81, 126, 50, 85, 126, 62, 1, 205, 1557
 7262 DATA 1, 22, 62, 22, 215, 62, 1, 215, 62, 17, 215, 58, 81, 126, 50, 1209
 7264 DATA 85, 126, 205, 44, 122, 201, 205, 126, 121, 58, 78, 126, 33, 81, 126, 1737
 7266 DATA 190, 208, 50, 80, 126, 205, 75, 119, 62, 22, 215, 62, 1, 215, 62, 1692
 7268 DATA 12, 215, 58, 80, 126, 50, 85, 126, 195, 44, 122, 205, 226, 123, 33, 1700
 7270 DATA 142, 123, 205, 11, 124, 205, 226, 115, 254, 76, 40, 121, 254, 108, 40, 2044
 7272 DATA 117, 254, 83, 202, 196, 123, 254, 115, 202, 196, 123, 24, 233, 22, 21, 2165
 7274 DATA 1, 80, 79, 78, 32, 69, 76, 32, 67, 65, 83, 83, 69, 84, 69, 967
 7276 DATA 32, 69, 78, 32, 77, 65, 82, 67, 72, 65, 46, 0, 0, 22, 20, 727
 7278 DATA 1, 80, 79, 78, 32, 82, 69, 67, 79, 82, 68, 32, 69, 78, 32, 928
 7280 DATA 69, 76, 32, 67, 65, 83, 83, 69, 84, 69, 46, 13, 80, 85, 76, 997
 7282 DATA 83, 65, 32, 85, 78, 65, 32, 84, 69, 67, 76, 65, 46, 0, 0, 847
 7284 DATA 22, 8, 10, 83, 61, 32, 83, 65, 76, 86, 65, 82, 46, 22, 12, 753
 7286 DATA 10, 76, 61, 32, 67, 65, 82, 71, 65, 82, 46, 0, 0, 205, 107, 969
 7288 DATA 13, 33, 65, 123, 205, 11, 124, 221, 33, 184, 136, 17, 0, 64, 62, 1291
 7290 DATA 128, 55, 205, 86, 5, 48, 232, 24, 88, 205, 107, 13, 33, 95, 123, 1447

7292 DATA 205, 11, 124, 205, 226, 115, 167, 40, 250, 221, 33, 184, 136, 17, 0, 1934
 7294 DATA 64, 62, 128, 55, 205, 194, 4, 24, 58, 17, 216, 214, 33, 0, 64, 1338
 7296 DATA 1, 0, 27, 237, 176, 205, 107, 13, 205, 83, 119, 62, 60, 50, 55, 1400
 7298 DATA 92, 201, 205, 226, 123, 33, 47, 124, 205, 11, 124, 205, 226, 115, 167, 2104
 7300 DATA 40, 250, 195, 28, 124, 126, 167, 40, 4, 215, 35, 24, 248, 35, 126, 1657
 7302 DATA 43, 167, 62, 0, 32, 244, 201, 62, 128, 50, 55, 92, 33, 216, 214, 1599
 7304 DATA 17, 0, 64, 1, 0, 27, 237, 176, 195, 83, 119, 20, 1, 77, 79, 1096
 7306 DATA 68, 79, 32, 78, 79, 82, 77, 65, 76, 58, 20, 0, 13, 45, 67, 839
 7308 DATA 65, 80, 83, 32, 83, 72, 73, 70, 84, 43, 49, 61, 73, 78, 83, 1029
 7310 DATA 84, 82, 85, 67, 67, 73, 79, 78, 69, 83, 46, 13, 45, 67, 65, 1003
 7312 DATA 80, 83, 32, 83, 72, 73, 70, 84, 43, 50, 61, 77, 65, 89, 85, 1047
 7314 DATA 83, 67, 46, 47, 77, 73, 78, 85, 83, 67, 46, 13, 45, 32, 32, 874
 7316 DATA 32, 32, 34, 32, 32, 32, 32, 32, 43, 51, 61, 77, 65, 82, 71, 708
 7318 DATA 69, 78, 32, 68, 69, 82, 69, 67, 72, 79, 46, 13, 45, 32, 32, 853
 7320 DATA 32, 32, 34, 32, 32, 32, 32, 32, 43, 52, 61, 77, 65, 82, 71, 709
 7322 DATA 69, 78, 32, 73, 90, 81, 85, 73, 69, 82, 46, 13, 45, 32, 32, 900
 7324 DATA 32, 32, 34, 32, 32, 32, 32, 32, 43, 57, 61, 83, 65, 86, 69, 722
 7326 DATA 47, 76, 79, 65, 68, 13, 45, 83, 46, 83, 72, 73, 70, 84, 43, 947
 7328 DATA 51, 61, 78, 8, 126, 13, 13, 20, 1, 77, 79, 68, 79, 32, 69, 775
 7330 DATA 88, 84, 69, 78, 68, 73, 68, 79, 58, 20, 0, 13, 45, 48, 61, 852
 7332 DATA 66, 79, 82, 82, 65, 32, 76, 73, 78, 69, 65, 32, 69, 78, 32, 978
 7334 DATA 67, 85, 82, 83, 79, 46, 13, 45, 49, 61, 76, 73, 78, 69, 65, 971
 7336 DATA 32, 65, 32, 76, 65, 32, 73, 90, 81, 85, 73, 69, 82, 68, 65, 988
 7338 DATA 46, 13, 45, 50, 61, 76, 73, 78, 69, 65, 32, 65, 32, 76, 65, 846
 7340 DATA 32, 68, 69, 82, 69, 67, 72, 65, 46, 13, 45, 51, 61, 73, 78, 891
 7342 DATA 83, 69, 82, 84, 65, 32, 76, 73, 78, 69, 65, 46, 13, 45, 52, 932
 7344 DATA 61, 73, 78, 83, 69, 82, 84, 65, 32, 67, 65, 82, 65, 67, 84, 1057
 7346 DATA 69, 82, 46, 13, 45, 53, 61, 67, 69, 78, 84, 82, 65, 32, 76, 922
 7348 DATA 73, 78, 69, 65, 46, 13, 45, 54, 61, 67, 85, 82, 83, 79, 82, 982
 7350 DATA 32, 65, 76, 32, 70, 73, 78, 65, 76, 46, 13, 45, 55, 61, 67, 854
 7352 DATA 85, 82, 83, 79, 82, 32, 65, 76, 32, 80, 82, 73, 78, 67, 73, 1069
 7354 DATA 80, 73, 79, 46, 13, 45, 56, 61, 66, 79, 82, 82, 65, 32, 67, 926
 7356 DATA 65, 82, 65, 67, 84, 69, 82, 46, 13, 45, 57, 61, 80, 65, 83, 964
 7358 DATA 65, 32, 80, 79, 82, 32, 73, 77, 80, 82, 69, 83, 79, 82, 65, 1060
 7360 DATA 46, 13, 45, 65, 44, 83, 44, 68, 44, 70, 44, 71, 61, 86, 79, 863
 7362 DATA 67, 65, 76, 69, 83, 32, 65, 67, 69, 78, 84, 85, 65, 68, 65, 1038
 7364 DATA 83, 46, 13, 13, 18, 1, 80, 85, 76, 83, 65, 32, 85, 78, 65, 823
 7366 DATA 32, 84, 69, 67, 76, 65, 32, 80, 65, 82, 65, 32, 86, 79, 76, 990
 7368 DATA 86, 69, 82, 46, 18, 0, 32, 0, 0, 67, 79, 76, 85, 77, 78, 795
 7370 DATA 65, 32, 32, 76, 73, 78, 69, 65, 32, 32, 67, 65, 80, 83, 32, 881
 7372 DATA 76, 46, 32, 77, 65, 82, 71, 46, 73, 90, 81, 46, 32, 77, 65, 959
 7374 DATA 82, 71, 46, 68, 69, 82, 46, 32, 77, 46, 69, 88, 84, 69, 78, 1007
 7376 DATA 68, 46, 32, 69, 68, 73, 84, 61, 73, 78, 83, 84, 46, 32, 0, 897
 7378 DATA 0, 0, 63, 0, 128, 0, 0, 0, 67, 46, 47, 77, 73, 78, 85, 664
 7380 DATA 83, 83
 9000 DATA 15, 1, 1, 1, 1, 1, 1, 1, 15, 36
 9002 DATA 15, 15, 15, 15, 0, 0, 0, 0, 60
 9004 DATA 240, 240, 240, 240, 0, 0, 0, 0, 960
 9006 DATA 255, 255, 255, 255, 0, 0, 0, 0, 1020
 9008 DATA 0, 0, 0, 0, 15, 15, 15, 15, 60
 9010 DATA 15, 15, 15, 15, 15, 15, 15, 15, 120
 9012 DATA 240, 240, 240, 240, 15, 15, 15, 15, 1020
 9014 DATA 255, 255, 255, 255, 15, 15, 15, 15, 1080
 9016 DATA 0, 0, 0, 0, 240, 240, 240, 240, 960
 9018 DATA 15, 15, 15, 15, 240, 240, 240, 240, 1020
 9020 DATA 240, 240, 240, 240, 240, 240, 240, 240, 1920
 9022 DATA 255, 255, 255, 255, 240, 240, 240, 240, 1980
 9024 DATA 0, 0, 0, 0, 255, 255, 255, 255, 1020
 9026 DATA 15, 15, 15, 15, 255, 255, 255, 255, 1080
 9028 DATA 240, 240, 240, 240, 255, 255, 255, 255, 1980
 9030 DATA 255, 255, 255, 255, 255, 255, 255, 255, 2040
 9032 DATA 60, 60, 60, 60, 60, 60, 60, 60, 480
 9034 DATA 255, 255, 0, 0, 0, 0, 255, 255, 1020
 9036 DATA 126, 126, 96, 96, 96, 96, 126, 126, 888
 9038 DATA 126, 126, 6, 6, 6, 6, 126, 126, 528
 9040 DATA 71, 121, 144, 71, 201, 45, 50, 1, 704
 9042 DATA 243, 205, 137, 248, 56, 11, 205, 101, 1206
 9044 DATA 250, 6, 64, 62, 32, 119, 35, 16, 584

9046 DATA 252, 58, 88, 249, 50, 1,
 243, 201, 1142
 9048 DATA 205, 220, 238, 216, 195,
 131, 246, 58, 1509
 9050 DATA 26, 243, 50, 1, 243, 201,
 205, 220, 1189
 9052 DATA 238, 216, 195, 59, 246,
 205, 220, 238, 1617
 9054 DATA 216, 195, 102, 246, 58,
 1, 243, 71, 1132
 9056 DATA 58, 29, 243, 61, 144,
 216, 58, 26, 835
 9058 DATA 243, 79, 120, 145, 201,
 33, 6, 243, 1070
 9060 DATA 203, 246, 205, 144, 225,
 201, 33, 6, 1263
 9062 DATA 243, 203, 182, 205, 144,
 225, 201, 32, 1435
 9064 DATA 0, 0, 0, 0, 0, 0, 0, 0, 0
 9066 DATA 0, 2, 2, 2, 2, 0, 2, 0, 10
 9068 DATA 0, 5, 5, 0, 0, 0, 0, 0, 10
 9070 DATA 3, 0, 6, 5, 5, 5, 5, 0, 29
 9072 DATA 0, 2, 7, 4, 7, 1, 7, 2, 30
 9074 DATA 0, 4, 4, 1, 2, 4, 1, 1, 17
 9076 DATA 0, 2, 5, 2, 6, 11, 15, 0, 41
 9078 DATA 0, 2, 4, 0, 0, 0, 0, 0, 6
 9080 DATA 0, 2, 4, 4, 4, 4, 2, 0, 20
 9082 DATA 0, 4, 2, 2, 2, 2, 4, 0, 16
 9084 DATA 0, 0, 5, 2, 7, 2, 5, 0, 21
 9086 DATA 0, 0, 2, 2, 7, 2, 2, 0, 15
 9088 DATA 0, 0, 0, 0, 0, 2, 2, 4, 8
 9090 DATA 0, 0, 0, 0, 7, 0, 0, 0, 7
 9092 DATA 0, 0, 0, 0, 0, 6, 6, 0, 12
 9094 DATA 0, 1, 1, 2, 2, 4, 4, 0, 14
 9096 DATA 0, 7, 5, 5, 5, 5, 7, 0, 34
 9098 DATA 0, 2, 6, 2, 2, 2, 7, 0, 21
 9100 DATA 0, 2, 5, 1, 2, 4, 7, 0, 21
 9102 DATA 0, 6, 1, 6, 1, 1, 6, 0, 21
 9104 DATA 0, 1, 3, 5, 5, 7, 1, 0, 22
 9106 DATA 0, 7, 4, 6, 1, 1, 6, 0, 25
 9108 DATA 0, 2, 4, 6, 5, 5, 2, 0, 24
 9110 DATA 0, 7, 1, 2, 2, 4, 4, 0, 20
 9112 DATA 0, 7, 5, 2, 5, 5, 7, 0, 31
 9114 DATA 0, 2, 5, 5, 3, 1, 6, 0, 22
 9116 DATA 0, 0, 0, 2, 0, 0, 2, 0, 4
 9118 DATA 0, 0, 2, 0, 0, 2, 2, 4, 10
 9120 DATA 0, 0, 1, 2, 4, 2, 1, 0, 10
 9122 DATA 0, 0, 0, 7, 0, 7, 0, 0, 14
 9124 DATA 0, 0, 4, 2, 1, 2, 4, 0, 13
 9126 DATA 0, 2, 5, 1, 2, 0, 2, 0, 12
 9128 DATA 0, 2, 0, 2, 2, 2, 0, 10
 9130 DATA 0, 7, 5, 5, 7, 5, 5, 0, 34
 9132 DATA 0, 7, 5, 6, 5, 5, 7, 0, 35
 9134 DATA 0, 7, 5, 4, 4, 5, 7, 0, 32
 9136 DATA 0, 6, 5, 5, 5, 5, 6, 0, 32
 9138 DATA 0, 7, 4, 6, 4, 4, 7, 0, 32
 9140 DATA 0, 7, 4, 7, 4, 4, 4, 0, 30
 9142 DATA 0, 7, 5, 4, 7, 5, 7, 0, 35
 9144 DATA 0, 5, 5, 7, 5, 5, 0, 32
 9146 DATA 0, 7, 2, 2, 2, 2, 7, 0, 22
 9148 DATA 0, 1, 1, 1, 5, 5, 7, 0, 20
 9150 DATA 0, 5, 5, 6, 6, 5, 5, 0, 32
 9152 DATA 0, 4, 4, 4, 4, 4, 7, 0, 27
 9154 DATA 0, 5, 7, 7, 7, 5, 5, 0, 36
 9156 DATA 0, 7, 5, 5, 5, 5, 0, 32
 9158 DATA 0, 7, 5, 5, 5, 5, 7, 0, 34
 9160 DATA 0, 7, 5, 5, 7, 4, 4, 0, 32
 9162 DATA 0, 7, 5, 5, 5, 7, 1, 37
 9164 DATA 0, 7, 5, 5, 6, 5, 5, 0, 33
 9166 DATA 0, 7, 4, 7, 1, 1, 7, 0, 27
 9168 DATA 0, 7, 2, 2, 2, 2, 0, 17
 9170 DATA 0, 5, 5, 5, 5, 5, 7, 0, 32
 9172 DATA 0, 5, 5, 5, 5, 2, 0, 27
 9174 DATA 0, 5, 7, 7, 7, 7, 2, 0, 35
 9176 DATA 0, 5, 5, 2, 2, 5, 5, 0, 24
 9178 DATA 0, 5, 5, 5, 2, 2, 0, 21
 9180 DATA 0, 7, 1, 2, 2, 4, 7, 0, 23
 9182 DATA 0, 7, 4, 4, 4, 4, 7, 0, 30
 9184 DATA 1, 2, 0, 6, 2, 2, 7, 0, 20
 9186 DATA 0, 7, 1, 1, 1, 1, 7, 0, 18
 9188 DATA 2, 2, 2, 2, 2, 2, 2, 16
 9190 DATA 0, 0, 0, 0, 0, 0, 15, 15
 9192 DATA 0, 2, 0, 2, 4, 5, 2, 0, 15
 9194 DATA 0, 0, 6, 1, 7, 5, 7, 0, 26
 9196 DATA 0, 4, 4, 6, 5, 5, 6, 0, 30
 9198 DATA 0, 0, 3, 4, 4, 4, 3, 0, 18
 9200 DATA 0, 1, 1, 3, 5, 5, 3, 0, 18
 9202 DATA 0, 0, 2, 5, 6, 4, 3, 0, 20
 9204 DATA 0, 3, 4, 6, 4, 4, 4, 0, 25
 9206 DATA 0, 0, 3, 5, 5, 3, 1, 6, 23
 9208 DATA 0, 4, 4, 6, 5, 5, 5, 0, 29
 9210 DATA 0, 2, 0, 6, 2, 2, 7, 0, 19
 9212 DATA 0, 1, 0, 1, 1, 1, 5, 2, 11
 9214 DATA 0, 4, 5, 6, 6, 5, 5, 0, 31
 9216 DATA 0, 4, 4, 4, 4, 4, 3, 0, 23
 9218 DATA 0, 0, 5, 7, 7, 7, 5, 0, 31
 9220 DATA 0, 0, 6, 5, 5, 5, 5, 0, 26
 9222 DATA 0, 0, 2, 5, 5, 5, 2, 0, 19
 9224 DATA 0, 0, 6, 5, 5, 6, 4, 4, 30
 9226 DATA 0, 0, 3, 5, 5, 3, 1, 1, 18
 9228 DATA 0, 0, 3, 4, 4, 4, 4, 0, 19
 9230 DATA 0, 0, 3, 4, 2, 1, 6, 0, 16
 9232 DATA 0, 2, 7, 2, 2, 2, 1, 0, 16
 9234 DATA 0, 0, 5, 5, 5, 5, 7, 0, 27
 9236 DATA 0, 0, 5, 5, 5, 5, 2, 0, 22
 9238 DATA 0, 0, 5, 7, 7, 7, 2, 0, 28
 9240 DATA 0, 0, 5, 5, 2, 5, 5, 0, 22
 9242 DATA 0, 0, 5, 5, 5, 3, 1, 6, 25
 9244 DATA 0, 0, 7, 1, 2, 4, 7, 0, 21
 9246 DATA 1, 0, 2, 5, 5, 5, 2, 0, 20
 9248 DATA 1, 0, 2, 5, 6, 4, 3, 0, 21
 9250 DATA 2, 0, 5, 5, 5, 5, 7, 0, 29
 9252 DATA 1, 0, 6, 1, 7, 5, 7, 0, 27
 9254 DATA 7, 0, 7, 5, 5, 5, 5, 0, 34
 9256 DATA 0, 0

TECNICAS DE ANALISIS

Sistemas y Subsistemas

ADA una de las etapas de desarrollo de la informatización se aplica a un campo de estudio específico, como hemos comentado (la empresa para el «esquema director», un dominio para el «estudio de viabilidad», una aplicación para el «estudio detallado»). La malla de análisis elegida en cada caso permite ir desde la visión global a los detalles de un modo progresivo y sin perder el punto de vista general, de modo que aseguremos la concepción y diseño de soluciones coherentes, de futuro e integradas.

Es importante tener claramente establecido el conjunto de herramientas conceptuales que forman esta retícula de análisis progresivo, para la eficacia de las soluciones propuestas. Hemos de considerar un conjunto de sistemas y subsistemas, en contraposición a los órganos físicos de la Organización (servicios, aplicaciones, medios informáticos, etc.).

En un primer nivel se consideran tres grandes «sistemas» en la estructura operativa (del mismo modo que se habla de «sistema nervioso», «sistema locomotor», etcétera, en el cuerpo humano; «sistema de frenada» o «sistema de alimentación», etcétera, en un automóvil): los tres sistemas ya aludidos son el «sistema de gestión» (donde se toman las decisiones sobre el desarrollo de las tareas de la empresa), «sistema de información» (que facilita datos al sistema de gestión e incor-

pora los canales de comunicación de la organización) y «sistema operacional» (que realiza las tareas cotidianas del funcionamiento de la empresa u organización). Sin embargo, hay que completar esta estructura del análisis en profundidad, mediante la definición de tres nociones adicionales: procedimiento, proceso y dominio.

— Un **procedimiento** es un conjunto de tareas que abordan una actividad concreta. El procedimiento se apoya sobre dos posiciones estables (llamadas «situaciones naturales», que no dependen de las soluciones existentes o por definir. Por ejemplo, la «llegada de una mercancía al almacén» (posición de «disparo» de la actividad y, por tanto, del procedimiento) provoca la apertura de los paquetes, comprobación del envío, comparación con los pedidos que en su día se hicieron, control de calidad, etc., hasta la vuelta al estado normal de reposo que se produce cuando la mercancía queda almacenada o es devuelta al proveedor. Como se ve, pueden participar varios órganos, personas, lugares físicos, etc., en el procedimiento: lo importante es asegurar la coherencia vertical entre los dos «puntos de anclaje» invariantes (un «suceso natural» y el «resultado natural»).

— Con el término **proceso** se designa un conjunto de procedimientos de la misma naturaleza. Por tanto, el proceso se encuentra articulado (como hemos comentado para el procedimiento) por dos «estados» estables «naturales» de la actividad de la organización y de su «sistema de información». La diferencia es que los límites del segmento de actividad que se examina no están marcados por dos sucesos, sino por toda una cadena de sucesos que se consideran relaciona-

dos en sentido horizontal: ahora los puntos de «anclaje» del proceso no vienen indicados por datos concretos, sino por «sucesos» designados de un modo mucho más genérico. (Véase el esquema adjunto.)

— Un **dominio** es un conjunto de procesos que, examinados desde un punto de vista más general, forman una unidad por referirse a un área coherente de actividad. La definición de esta «especie de coherencia» del dominio se basa en tres criterios:

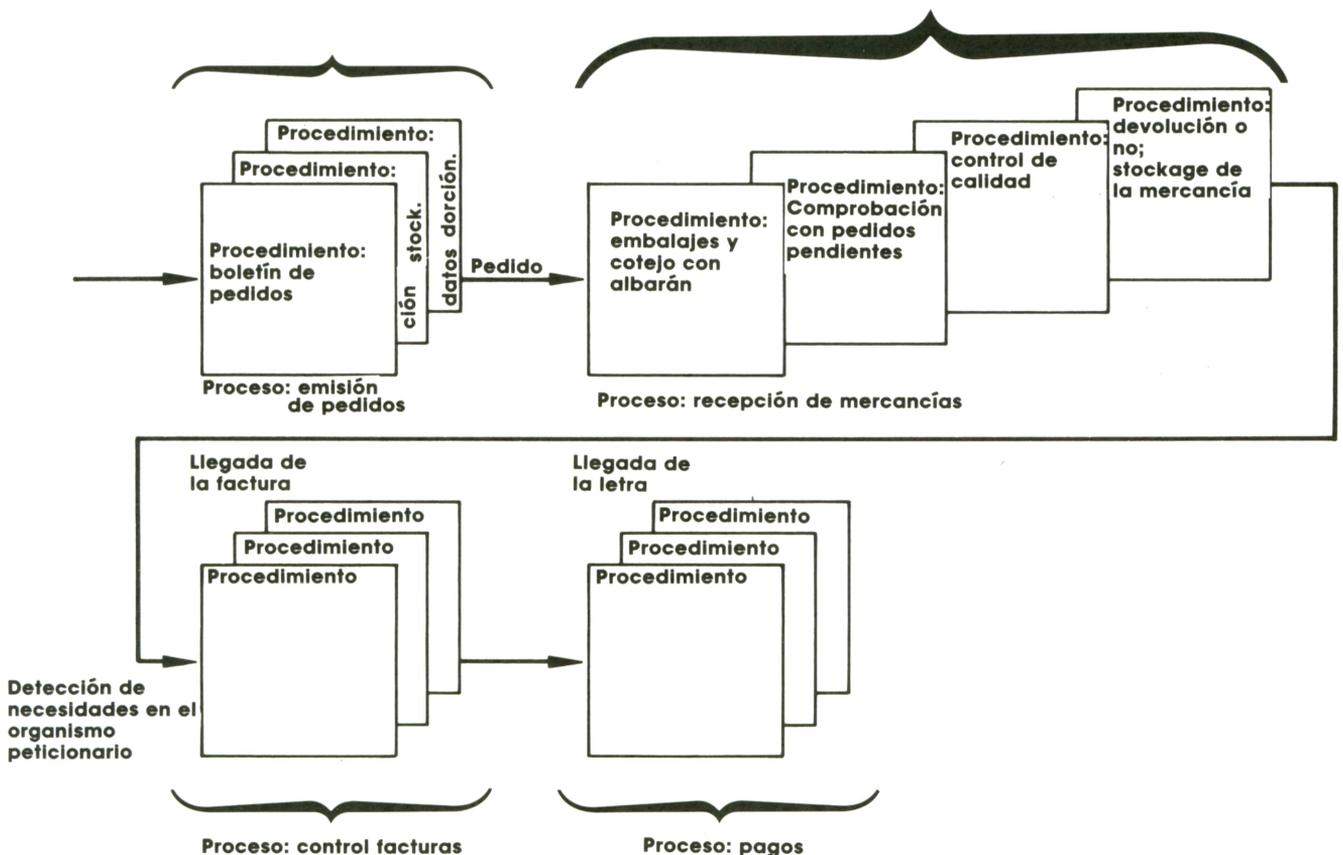
a) La *homogeneidad de las finalidades* de gestión de los procesos componentes. Esto aporta una gran dependencia cronológica entre dichos procesos.

b) La *homogeneidad de la estructura de gestión* de los procesos componentes del dominio. En efecto, la velocidad de convergencia de las ideas para concebir soluciones futuras, está supeditada en gran manera al número de personas de cuyas decisiones depende la definición de los objetivos, las orientaciones de la

actividad y los sistemas de arbitraje que haya que establecer.

c) *Coherencia de los elementos* manipulados en el *sistema de información* subyacente. Si la responsabilidad de puesta al día de una entidad del sistema de información está compartida entre varios dominios, es dudoso que se mantenga en el tiempo su integridad.

Respecto de la estructura de análisis definida al comienzo (y analizada ya anteriormente), hemos de decir que los dominios «surgen» en el establecimiento del «esquema director». En efecto, del análisis global de la organización surgen estos grandes dominios de aplicación e, incluso, los procesos relevantes de los dominios críticos. El estudio y definición de estos dominios y sus límites (así como de los procesos relevantes en ellos incluidos) aporta una visión básica imprescindible para el establecimiento de los «escenarios» globales de evolución de las actividades.



TECNICAS DE PROGRAMACION

Otras instrucciones de transferencia

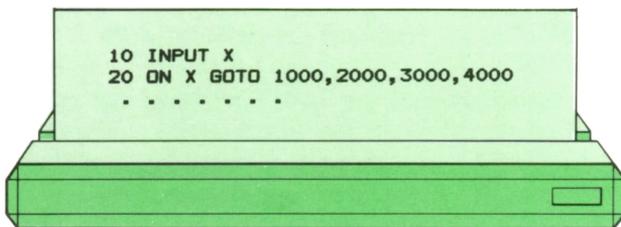
P

UESTO que en el lenguaje PASCAL la instrucción de transferencia debe utilizarse sólo excepcionalmente, no existen otras variantes de este tipo de instrucciones.

Sin embargo, en el lenguaje BASIC, donde las instrucciones de transferencia son muy frecuentes, existe otro tipo de éstas, bastante útil, que vamos a ver a continuación y que reciben el nombre de «instrucciones de transferencia calculada». Estas instrucciones tienen la siguiente forma:

ON expresión GOTO lista de etiquetas.

donde ON es la palabra inglesa que significa «sobre», GOTO significa, como siempre, «ir a», «expresión» es una expresión BASIC cualquiera de resultado entero y «lista de etiquetas» es un conjunto de etiquetas separadas por comas. Veamos un ejemplo:



El programa anterior lee del teclado el valor de X. A continuación, la línea 20 transfiere el control a la línea 1000 si el valor de X era igual a 1; a la línea 2000, si X es igual a 2; a la línea 3000, si es igual a 3; y a la línea 4000 si es igual a 4. En

cualquier otro caso (para cualquier otro valor entero de X, incluso negativo) no se realizará ninguna transferencia y la ejecución continuará en la instrucción siguiente a la de etiqueta 20.

El organigrama del programa anterior es el siguiente:

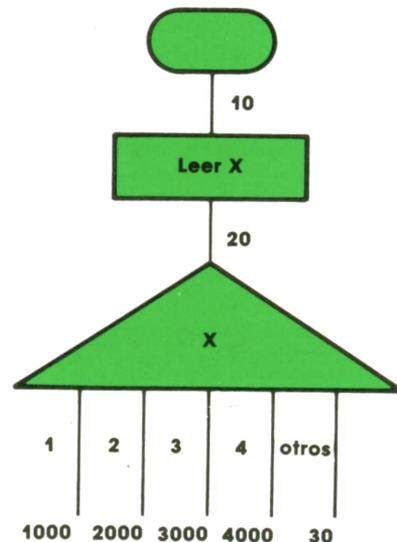
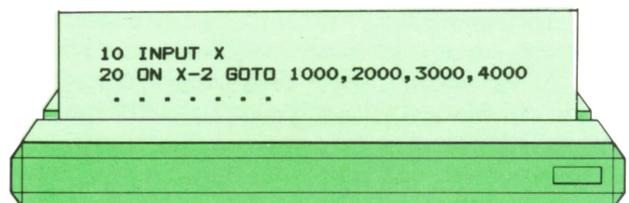


Figura 1

En lugar de X podríamos haber utilizado otra expresión, como en el caso siguiente:



cuyo organigrama es:

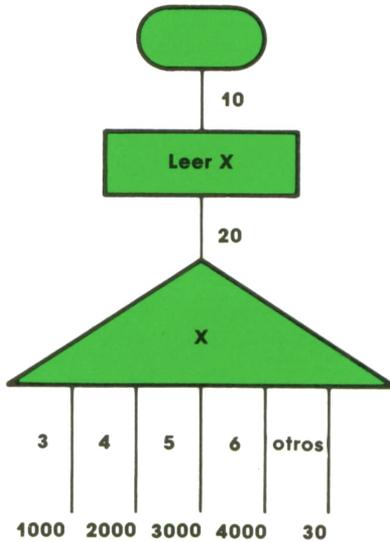


Figura 2



Instrucciones de transferencia en el lenguaje APL

En el lenguaje APL, la instrucción de transferencia es la única que permite variar la marcha de la ejecución de los programas, haciéndola no secuencial, pues no hay instrucciones explícitas de bucle. Se recordará que las etiquetas APL son nombres ordinarios separados de la instrucción por dos puntos (:).

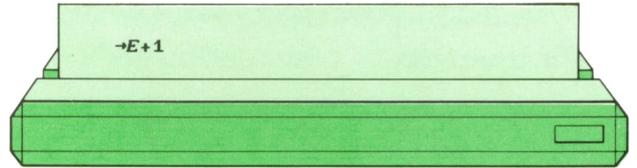
La transferencia incondicional APL se escribe anteponiendo al nombre de la etiqueta una flecha dirigida hacia la derecha, que se lee «ir a». Por ejemplo:



La instrucción anterior da lugar a una transferencia incondicional a la instrucción cuya etiqueta es E.

Supongamos que un programa APL tiene 10 líneas. En tal caso, el intérprete de APL supone que estas líneas están numeradas automáticamente de 1 a 10. Si la línea número 4 tiene una etiqueta (por ejemplo, E), durante la ejecución del programa se crea una variable temporal llamada E, cuyo valor es igual al número

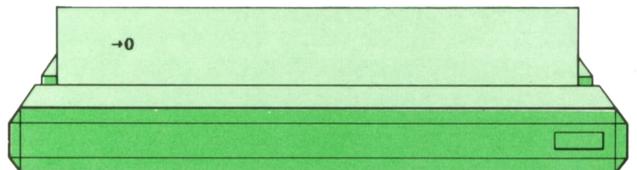
de línea que tiene dicha etiqueta (en este caso, 4). Esta variable se comporta del mismo modo que cualquier otra variable numérica. En particular, pueden hacerse operaciones con ella. Por ejemplo, la instrucción:



transfiere a la instrucción siguiente a la que tiene la etiqueta E, mientras que la instrucción

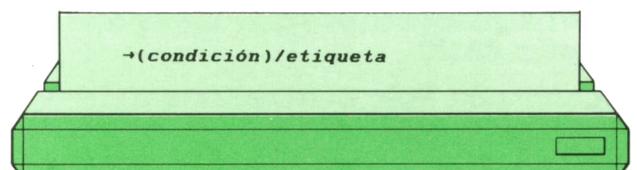


transfiere a la etiqueta anterior. Naturalmente, es posible que el resultado de la operación no corresponda a una línea del programa. Podría ser cero, un número entero negativo, o un número positivo mayor que el número de líneas del programa completo (en nuestro caso, 10). Si esto ocurre, el intérprete da por terminada la ejecución de este programa, pasando al que lo llamó o terminando totalmente la ejecución. Por tanto, la forma típica de abandonar la ejecución de un programa en APL es la instrucción:



La facilidad de realizar operaciones con las etiquetas da una flexibilidad enorme a las instrucciones de transferencia del APL, que pueden ajustarse al gusto del programador y realizar con gran sencillez operaciones aparentemente muy complicadas. Veamos algunos ejemplos:

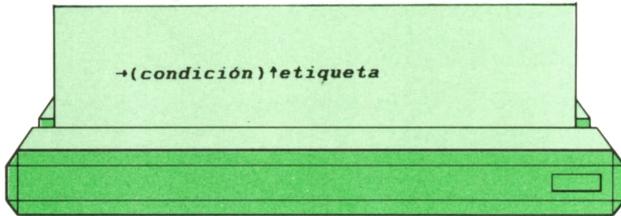
La instrucción



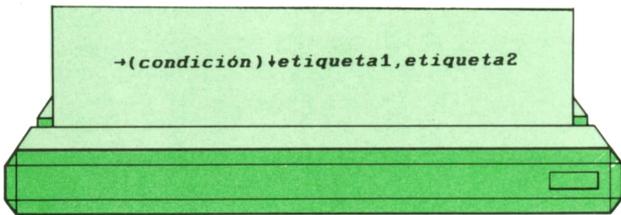
transfiere el control la instrucción que tiene la etiqueta indicada si la condición se cumple, pero continúa con la instrucción siguiente si la condición no se cumple. Es, por tanto, equivalente a la instrucción BASIC

IF condición THEN GOTO etiqueta

La instrucción



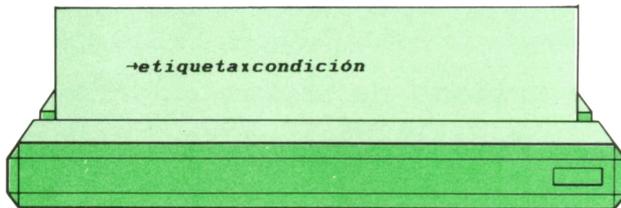
tiene exactamente el mismo efecto.
La instrucción



transfiere el control a la instrucción llamada etiqueta2 si la condición se cumple, y la instrucción llamada etiqueta1 si la condición no se cumple. Es, por tanto, equivalente a la instrucción BASIC

**IF condición THEN GOTO etiqueta2
ELSE GOTO etiqueta1**

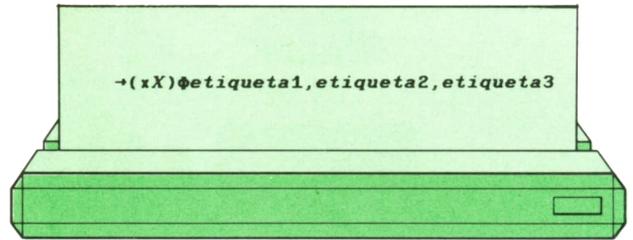
La instrucción:



transfiere el control a la instrucción llamada etiqueta si la condición se cumple, y abandona la ejecución del programa si la condición no se cumple. Es, por tanto, en algunos casos equivalente a la instrucción BASIC

**IF condición THEN GOTO etiqueta
ELSE RETURN**

La instrucción



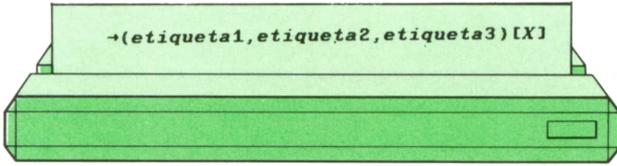
transfiere el control a la instrucción llamada etiqueta1 si X es igual a cero, a la instrucción llamada etiqueta2 si X es positivo y a la instrucción llamada etiqueta3 si X es negativo. Es, por tanto, equivalente a la instrucción BASIC

**IF X = 0 THEN GOTO etiqueta1
ELSE IF X > 0 THEN GOTO etiqueta2
ELSE GOTO etiqueta3**

Veamos por qué. El signo de multiplicar aplicado a una variable es, en APL, la función signo, que nos da el resultado cero si X es igual a cero; el resultado 1 si X es positivo, y el resultado -1 si X es negativo. La operación representada por la letra griega «fi», aplicada a una serie (en nuestro caso etiqueta1, etiqueta2, etiqueta3) lo rota tantas posiciones como indique el número situado a la izquierda de la letra «fi» (que, en nuestro caso, es el resultado de «signo de X», o sea 0, 1 o -1). Por tanto, si X es cero, la serie se rota cero posiciones y se queda como estaba: etiqueta1, etiqueta2, etiqueta3. La transferencia se efectúa, por tanto, a etiqueta1, que es el primer elemento de la serie.

Si X es positivo, hay que rotar una posición los elementos de la serie (de derecha a izquierda), con lo que se obtiene la nueva serie etiqueta2, etiqueta3, etiqueta1, y la transferencia se efectúa a etiqueta2. Finalmente, si X es negativo, hay que rotar los elementos de la serie -1 posiciones hacia la izquierda, es decir, una posición hacia la derecha, con lo que la serie queda etiqueta3, etiqueta1, etiqueta2 y la transferencia se efectúa, como siempre, al primer elemento (etiqueta3).

Vemos un último ejemplo de entre las infinitas formas que puede adoptar en APL la instrucción de transferencia. (Por supuesto, no es preciso aprenderlas todas, sino sólo las que vayamos a utilizar con más frecuencia).



Esta instrucción transfiere el control a la instrucción llamada etiqueta1 si X es igual a 1, a la instrucción llamada etiqueta2 si X es igual a 2, y a la instrucción llamada etiqueta3 si X es igual a 3. Es, por tanto, equivalente a la instrucción BASIC

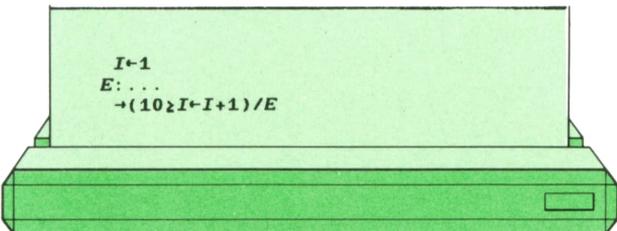
ON X GOTO etiqueta1, etiqueta2, etiqueta3

Para terminar con las instrucciones de transferencia, vamos a ver cómo se programan en APL diferentes clases de bucles. En primer lugar, el que en BASIC se escribiría así:

FOR I = 1 TO 10

**..
NEXT I**

El programa correspondiente es:



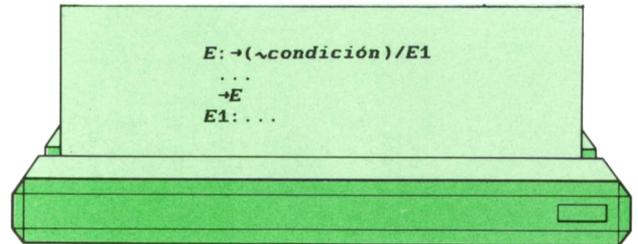
Es decir: se asigna a I el valor inicial; se ejecuta el bucle; se incrementa en una

unidad el valor de I y se compara con el valor final (10). Si es menor o igual, hay que seguir ejecutando el bucle y transferimos a la etiqueta E. En caso contrario, seguimos con la instrucción siguiente.

En segundo lugar, el bucle BASIC **WHILE condición**

**..
WEND**

El programa correspondiente es:



Es decir, en primer lugar, se comprueba la condición. Si no se cumple, se abandona el bucle (el signo representado por una tilde es la negación lógica). Si se cumple, se ejecuta el interior del bucle y se vuelve a la etiqueta E (principio del bucle) para volver a comprobar la condición.

Es evidente que esta forma de programar da al programador una libertad absoluta y que pueden construirse estructuras de control y bucles tan complejos como se quiera, incluso muchos que no existen como tales en BASIC o PASCAL y que sólo podrían programarse con ayuda de las instrucciones de transferencia.

APLICACIONES

HOJAS DE CALCULO LOTUS 1-2-3

Un ejemplo

A

fin de llegar a conocer la hoja de cálculo vamos a pasar a ilustrar los puntos previos con un ejemplo:

El problema se plantea en unos almacenes de un comercio,

donde precisan llevar listas con datos de los artículos a cada sucursal, indicándoles el descuento a realizar en cada caso si éste es posible.

D12: [A11] * PERC. VENTA						ACTIV.	
	A	B	C	D	E	F	
1							
2	DESCUENTO DE PRIMAVERA						
3	ARTICULO	PERC. INICIAL	COSTE	PERC. VENTA	MARGEN		
4	DISCOS						
5	ORD PERSONALES						
6	YOCADISCOS						
7	ARRIBOS						
8							
9							
10							
11	DESCUENTO VERANO						
12	ARTICULO	PERC. INICIAL	COSTE	PERC. VENTA	MARGEN		
13	DISCOS						
14	ORD PERSONALES						
15	YOCADISCOS						
16	ARRIBOS						
17							
18							
19							
20							
19/05/81 11:05						CAPS	

Observe la posibilidad de crear una línea que separe el encabezamiento de los

A14: [A14] * PERC. VENTA						ACTIVO	
	A	B	C	D	E	F	
1							
2	DESCUENTO DE PRIMAVERA						
3	ARTICULO	PERC. INICIAL	COSTE	PERC. VENTA	MARGEN		
4	DISCOS						
5	ORD PERSONALES						
6	YOCADISCOS						
7	ARRIBOS						
8							
9							
10							
11	DESCUENTO VERANO						
12	ARTICULO	PERC. INICIAL	COSTE	PERC. VENTA	MARGEN		
13	DISCOS						
14	ORD PERSONALES						
15	YOCADISCOS						
16	ARRIBOS						
17							
18							
19							
20							
19/05/81 11:05						CAPS	

Comenzamos por crear la máscara de la hoja de trabajo introduciendo el texto. Sitúese en la celda A1 y comience a teclear el encabezamiento. Observará cómo el indicador de modalidad cambia de ACTIVO a ROTULO.

Cuando termine el rótulo completo puede pulsar (Intro) o cualquier tecla del cursor. Observará que a pesar de ser de longitud mayor a la de las celdas podemos visualizar todo el contenido de la misma, ya que ésta se posiciona sobre las celdas contiguas. Esto sólo ocurre en el caso de los rótulos y sólo mientras en la celda de la derecha no exista otro contenido. En nuestro caso como se trata del encabezamiento la situación es válida.

Continúe tecleando los rótulos hasta que estén todos.

rótulos laterales. Para ello vamos a insertar una línea en blanco y después introducimos la separación, repitiendo el proceso para el descuento de verano. En este caso, una vez insertada la línea copiamos la del caso anterior.

Una forma de escribir la línea para subrayar es como se ha hecho en la figura, es decir, utilizando una sola celda para escribir una larga tira de signos «=». Otro método, algo más cómodo, es utilizando la barra inclinada a la izquierda (\) y después el símbolo que queramos que aparezca repetido, por ejemplo: \=.

Proceda a grabar la máscara que acaba de realizar.

A19: [A19] '..... EDICION
 Introduzca nombre de fichero a grabar: A:ASCARA
 A B C D E F
 1
 2 DESCUENTO DE PRIMAVERA 101
 3 ARTICULO PREC.INICIAL COSTE PREC.VENTA MARGEN
 4
 5 DISCOS
 6 ORO PERSONALIAS
 7 TOCADISCOS
 8 AMEIOS
 9
 10
 11
 12 DESCUENTO DE VERANO 251
 13 ARTICULO PREC.INICIAL COSTE PREC.VENTA MARGEN
 14
 15 DISCOS
 16 ORO PERSONALIAS
 17 TOCADISCOS
 18 AMEIOS
 19
 20
 19/05/87 11:13 CAPS

El siguiente paso consiste en la introducción de las fórmulas:

En primer lugar, introducimos el descuento a realizar en cada caso. Este debe ir expresado en tanto por ciento, para lo cual debemos acudir al comando RANGO; una vez en él, seleccionamos FMTO, y en éste seleccionamos la opción de que se exprese en %. Pulsamos la tecla (Intro) y Lotus nos pregunta el número de decimales deseado, en este caso respondemos 0. Sólo queda indicar el rango de celdas que deben cumplir este formato: D2 y posteriormente con COPIAR lo pasamos a D12.

D2: (P0) [A11] 0,1
 Fijo Científico Monetario . General +/- I Día Texto OculLa Restablecer
 Permito porcentaje (x,xx)
 A B C D E F
 1
 2 DESCUENTO DE PRIMAVERA 101
 3 ARTICULO PREC.INICIAL COSTE PREC.VENTA MARGEN
 4
 5 DISCOS
 6 ORO PERSONALIAS
 7 TOCADISCOS
 8 AMEIOS
 9
 10
 11
 12 DESCUENTO DE VERANO 251
 13 ARTICULO PREC.INICIAL COSTE PREC.VENTA MARGEN
 14
 15 DISCOS
 16 ORO PERSONALIAS
 17 TOCADISCOS
 18 AMEIOS
 19
 20
 19/05/87 11:30 CAPS

Pasamos ya a las fórmulas en sí. En primer lugar, planteamos el precio de venta, que será:

$$\text{PREC.VENTA} = \text{PREC.INICIAL} - (\text{PREC.INICIAL} * \text{DESCUENTO})$$

Nos situamos en la celda D5 y tecleamos:

$$+B5-(B5*\$D\$2)$$

\$D\$2 hace una referencia absoluta de la celda D2, de forma que al copiar la fórmula en las líneas siguientes las celdas relativas son tratadas como tales y D2 permanece fija en la fórmula.

Será necesario cambiar esta referencia absoluta para copiar las fórmulas del descuento de verano poniendo: \$D\$12.

La siguiente fórmula a tratar es el margen obtenido:

$$\text{MARGEN} = \text{PREC.VENTA} - \text{COSTE}$$

Nos situamos en la celda E5 y tecleamos:

$$+D5-C5$$

Nuevamente copiamos a las líneas siguientes:

DS: [A11] +B5-(B5+10A2) ACTIVO

	A	B	C	D	E	F
1						
2	DESCRITO DE PRIMAVERA			101		
3	ARTICULO	PREC. INICIAL	COSTE	PREC. VENTA	MARGEN	
4					
5	DISCOS		0	0		
6	ORD PERSONALES		0	0		
7	TOCADISCOS		0	0		
8	AMBIOS		0	0		
9						
10						
11						
12	DESCRITO DE VERANO			251		
13	ARTICULO	PREC. INICIAL	COSTE	PREC. VENTA	MARGEN	
14					
15	DISCOS		0	0		
16	ORD PERSONALES		0	0		
17	TOCADISCOS		0	0		
18	AMBIOS		0	0		
19						
20						

19/05/87 11:39 CAPS

ES: +B16-C18 ACTIVO

	A	B	C	D	E	F
1						
2	DESCRITO DE PRIMAVERA			101		
3	ARTICULO	PREC. INICIAL	COSTE	PREC. VENTA	MARGEN	
4					
5	DISCOS	1000	250	900	650	
6	ORD PERSONALES	150000	50000	135000	85000	
7	TOCADISCOS	80000	25000	72000	47000	
8	AMBIOS	50000	15000	45000	30000	
9						
10						
11						
12	DESCRITO DE VERANO			251		
13	ARTICULO	PREC. INICIAL	COSTE	PREC. VENTA	MARGEN	
14					
15	DISCOS	1000	250	750	500	
16	ORD PERSONALES	150000	50000	112500	62500	
17	TOCADISCOS	80000	25000	60000	35000	
18	AMBIOS	50000	15000	37500	22500	
19						
20						

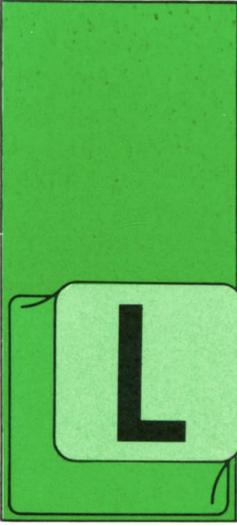
19/05/87 11:45 CAPS

Pasamos al siguiente paso, que consiste en la grabación del modelo creado hasta el momento y la posterior introducción de los datos para verificar que éste es válido.

Se comprueban los resultados obtenidos, si éstos son correctos ya se puede utilizar la hoja de cálculo para el trabajo concreto que queríamos realizar.

PASCAL

LA INSTRUCCION GOTO



A instrucción GOTO (go to, «ir a», en inglés) hace que se continúe ejecutando el programa en otro punto distinto a aquél en que se encuentra en ese momento. Por

decirlo de otra manera, permite «saltar» a otro punto del programa desde el lugar en que se encuentra la instrucción. Es equivalente a la instrucción GOTO de otros lenguajes como BASIC o FORTRAN.

El PASCAL tiene todas las estructuras de control necesarias para construir un programa, es decir, secuencias, bifurcaciones y repeticiones, por lo que en multitud de ocasiones se le ha descrito de manera pobre y superficial como un «lenguaje para programar sin GOTO».

Sin embargo, hay casos (pocos) en que la utilización de GOTO puede simplificar un programa y por ello se contempla su uso en PASCAL. Estos casos suelen

ser aquéllos en que, por algún suceso extraordinario, se desea cambiar la marcha normal de un programa. En cualquier caso, dada su gran potencia, debe utilizarse con mucho cuidado y sólo en casos muy especiales.

La instrucción consta de la palabra reservada GOTO seguida de la «etiqueta» del punto al que se desea transferir la ejecución del programa. Esta etiqueta puede ser cualquier número natural de cuatro cifras como máximo, aunque algunas versiones de PASCAL permiten también el empleo de palabras o identificadores válidos. La etiqueta debe estar escrita justo antes de la instrucción a la que se desea saltar separada de ella por dos puntos.

Tan excepcional se considera su utilización que todas las etiquetas que se necesitan deben ser declaradas previamente, tras la cabecera del programa (o procedimiento) y antes de la definición de datos, de la siguiente manera:

```

program EjemploGOTO;

  label 10,20;      (* Label significa etiqueta *)

  const Pi = 3.141592654;

  var   N : integer;

begin
  N := 0;

10: writeln ('Esto se escribe repetidas veces. ');
   N := N + 1;
   if N < 4 then goto 10;

   goto 20;
   writeln ('Pero esto ni una. ');
20: writeln ('Adiós. ');
end.

```

es decir, tras la palabra reservada LABEL se escriben las diferentes etiquetas separadas entre sí por comas. Para terminar se escribe un punto y coma.

Caso típico de utilización de GOTO es

aquél en que se detecta un error en una fase temprana de la ejecución de un programa y se desea entonces que se detenga. Para conseguir esto sin GOTO se podría hacer:

```

.....
Error1:= (...la condición de error que sea...)
if not Error1 then (* ejecutar el resto del programa: *)
begin
.....
Error2 := (...la condición de error que sea...)
if not Error2 then
begin
.....
end
end;
writeln ('Fin del programa.')
end.

```

o sea, poner como condición para la ejecución de lo que viene a continuación la no existencia de error. Si, por ejemplo, el error se detectase dentro de una estructura REPEAT, habría que ponerlo también como condición de salida del bucle:

repeat

.....

Error3 := (... la condición que sea)
until Error3 or Salir

Sin embargo, mediante GOTO la cuestión se simplifica:

.....

If Error1 then goto 100;

.....

If Error2 then goto 100;

.....

repeat

.....

if Error3 then goto 100
until Salir;

.....

100: writeln ('Fin del programa'.)
end.

La única restricción existente en el PASCAL estándar a la hora de realizar un salto es que sólo se puede saltar a un punto dentro del mismo bloque de programa en que nos encontremos en el momento del salto, es decir, no se puede ir desde dentro de un procedimiento a otro que sea independiente de aquél o de nivel inferior.

A la hora de buscar la etiqueta de destino de una instrucción GOTO se empieza mirando en las definidas localmente, pasándose luego a buscar entre las del procedimiento en que esté inserto el propietario del GOTO, etc., de manera análoga a como se hace con las variables o con las llamadas a procedimientos. En otras palabras, desde una instrucción de un subprograma dado sólo se puede saltar a otra instrucción del mismo subprograma, o de un subprograma en el que esté inserto aquél (incluyéndose, por tanto, la zona de instrucciones del programa principal) y siempre que esa misma etiqueta no esté definida localmente.

El salto al interior de una instrucción estructurada (IF, FOR...) desde fuera de ella puede producir errores inesperados, aunque hay compiladores que no avisan de la presencia de semejantes situaciones. Veamos unos ejemplos de empleo incorrecto de GOTO:

.....

```
procedure Fase0;
  label 10;
  begin
    10: writeln
  end;
```

```
goto 10;
(* Salto inválido pues la etiqueta *)
(* 10 es un detalle interno de Fase0 *)
(* y aquí no se conoce su existencia *)
```

```
for I := 1 to 10 do
  begin
    Fase1;
    3: Fase2
  end;
goto 3;
```

```
goto 20;
if A then 20: writeln;
```

.....

Puede que alguien sienta la tentación de declarar al principio:

```
label
  10, 20, 30...
  ...500, 510, 520...
  ...1000, 1010, 1020...;
```

y programar «al estilo BASIC»; en ese caso es que no ha comprendido el propósito y las ventajas de la programación estructurada. Sin embargo, tampoco se debe ser purista y complicarse la vida para no utilizar jamás la instrucción GOTO. La norma que hay que seguir en todo momento es la de máxima claridad y, si ésta se consigue con GOTO, no hay que dudar en utilizarla.

NOTAS:

— La utilidad de acabar la ejecución de un procedimiento mediante un salto al final es tal, que muchos compiladores disponen de una instrucción de salto especial para ello, que suele escribirse como EXIT (salida).

— Existen algunos compiladores que son más restrictivos que el PASCAL estándar, permitiendo solamente el salto entre puntos pertenecientes a un mismo subprograma.



Almacenamiento dinámico

Con lo que sabemos por ahora, cuando hemos necesitado guardar en memoria muchas fichas para, por ejemplo, ordenarlas según un criterio dado, hemos acudido a estructuras de tipo «array of record».

Este tipo de almacenamiento se denomina estático, pues la porción de memoria destinada a las fichas se asigna al escribir el programa y no cambia durante su ejecución. Hasta ahora, todos los ejemplos de almacenamiento en memoria de variables globales que hemos visto han utilizado almacenamiento estático.

Existe, no obstante, un método de almacenamiento en memoria denominado «dinámico», que permite reservar porciones de memoria sobre la marcha e incluso utilizar una determinada zona para diferentes cometidos en diferentes momentos. Aunque las variables locales de un procedimiento tienen estas características, sólo existen durante su ejecución y, además, su número y la cantidad de memoria que ocupan sí está definida al escribir el programa, por lo que, para evitar confusiones, no las incluiremos entre las que llamaremos desde ahora variables «dinámicas».

El programa de ordenación de datos que escribimos en otra ocasión tenía el inconveniente de que el máximo número de fichas (100) estaba definido al crear el programa. En casos así, para evitar quedarse corto hay que crear tablas suficientemente grandes, pero entonces habrá ocasiones en que sólo se utilice una pequeña parte de la memoria reservada al principio.

En PASCAL es posible crear variables, no en el momento de escribir el programa, sino cuando éste se está ejecutando y a medida que se vayan necesitando. Al no estar definidas en la zona de declaración de datos, estas variables no tienen nombre y para referirse a ellas se utilizan los denominados PUNTEROS.

Un puntero es una variable especial que sirve para guardar una indicación de en qué sitio de la memoria se encuentra una variable dinámica. Si escribimos:

```

type
  Nombre_t = array (1..12) of char;
  Ficha_t = record
    Nombre,
    Apellido1,
    Apellido2 : Nombre_t;
    Nota_A,
    Nota_B,
    Nota_C : real
  end;

  Punt_t = ^Ficha_;
var
  A, B: Punt_t;

```

le estamos diciendo al compilador que todas las variables de tipo Punt_t son punteros que sirven para «apuntar» a variables de tipo Ficha_t. Es decir, el tipo de puntero se indica con el símbolo ^ seguido del tipo de variable al que apunta.

Cuando el programa ya está funcionando, para reservar sitio para una nueva ficha se utiliza la función NEW (nuevo en inglés):

new (A);

de esta manera se reservaría memoria para una variable de tipo Ficha_t y su dirección quedaría guardada en la variable A. Cuando quisiéramos utilizar la ficha, en lugar del nombre que no tiene pondríamos A^, que significa «la variable apuntada por A» y que es totalmente equivalente.

La única operación posible entre punteros es la asignación, es decir, guardar el contenido de uno en otro.

Para guardar muchas fichas necesitaríamos tener tantos punteros como fichas, por ejemplo, con un «array of Punt_t», con lo que si éste se llamara Tabla, para leer las fichas podríamos hacer:

```

for I := 1 to Total do
  begin
    (* Reservar sitio: *)
    new (Tabla [I]);
    (* Rellenarla: *)
    with Tabla [I]^ do
      begin

```

```

.....
end

```

end;

La modificación del programa Ordenar resultaría muy fácil; el procedimiento Ordena, no obstante, podría ser mejorado ligeramente, pues para permutar dos fichas bastaría con permutar sus punteros sin tocarlas a ellas para nada.

Esta manera de proceder sigue teniendo el mismo problema que la anterior, aunque menos grave, pues, al ocupar un puntero mucha menos memoria que una ficha, nos podríamos permitir preparar la tabla con un número sobradamente amplio de elementos.

Una solución mejor podría ser guardar el puntero de cada ficha en la anterior a ella, en un campo especialmente preparado para ello.

Para llegar a una ficha dada habría que tomar de la primera el puntero que alberga y con él podríamos utilizar ya la segunda ficha, de la que tomaríamos el puntero que lleva a la tercera, etc., hasta encontrar la ficha deseada.

De esta manera, tanto punteros como fichas se irían reservando en memoria según se fueran necesitando. Las fichas quedarían más o menos así:

Alvaro Cavero Comino	Ernesto Fernández González	Carlos González Pozas
7.2	8.9	7.6
5.1	7.3	9.1
4.5	7.6	8.5
(SIGUIENTE)	(SIGUIENTE)	(SIGUIENTE)

etc.

Esto es lo que se denomina una estructura de tipo «lista encadenada». Haría falta además un puntero aparte para poder utilizar el primer elemento. Para indicar que un elemento de la lista es el último lo que se hace es dar al puntero que alberga el valor predefinido NIL, cuyo significado es que no apunta a ninguna variable; por ejemplo:

Siguiente := nil;

FIN

OTROS LENGUAJES

FORTRAN

Sentencias de entrada/salida

AS sentencias de e/s se utilizan para realizar el intercambio de información entre el ordenador y los periféricos. En este apartado sólo se define la entrada/salida reali-

zada a través de la pantalla y del teclado.

Para un compilador FORTRAN cada uno de los dispositivos tiene asociado un número lógico. En el FORTRAN utilizado son:

- 0: para la entrada por pantalla.
- 0: para la salida por pantalla.

Estos números aparecen en todas las instrucciones para identificar el periférico. El formato de estas instrucciones es:

Lectura: `READ (0, NUM1) variable-1, variable-2, ..., variable-n`

Escritura: `WRITE (0, NUM2) variable-1, variable-2, ..., variable-n`

que permiten leer y escribir la lista de variables que las acompañan.

Para saber en qué forma se debe hacer la operación, las instrucciones de e/s van acompañadas de una sentencia declarativa: sentencia `FORMAT`, identificada por un número de sentencia (`NUM1`, `NUM2`) que se corresponde con el número de formato especificado en la sentencia de e/s.

```
READ (0,100) MES,PESO
100 FORMAT ( ... )
```

Las sentencias `FORMAT` pueden aparecer en cualquier parte del programa, aunque para mejorar la legibilidad del mismo es conveniente situarlas bien detrás de la operación asociada, bien todas juntas al final del programa.

Dentro de la sentencia `FORMAT` pueden aparecer los caracteres:

“/”: Indica que la lectura o escritura debe realizarse en la siguiente línea. Pueden utilizarse repetidas y no precisan estar separadas por comas.

nX: Deja n espacios en blanco.

«1»: Si se emplea con una impresora, hace que salte a la página siguiente.

```
PROGRAMA CON FORMATOS DE EDICION.
```

```
      WRITE ( 0, 100 )
100   FORMAT ( 5X, 'ESTE ES UN EJEMPLO DE ESCRITURA' )
      WRITE ( 0, 200 )
200   FORMAT ( 9X, 'ESCRIBO ESTA FRASE DESPLAZADA' )
      WRITE ( 0, 300 )
300   FORMAT ( //, 1X, 'AHORA SALTO UNA LINEA' )
      STOP
      END
```

Para leer o escribir una variable entera se utiliza la especificación I_n , donde n indica el número de caracteres (longitud del campo) reservados para la variable. Si n es menor que la longitud de la variable, se truncará, y si es mayor se rellenará con ceros a la izquierda.

Si la variable tiene signo es necesario reservar una posición más. Si en la lectura se dejan espacios en blanco en los últimos caracteres, el FORTRAN los interpreta como ceros.

$F_n.d$, sirve para leer o escribir variables reales. nx es la longitud total del dato, incluyendo el punto y el signo. d es el número de cifras decimales.

Si se trata de una variable lógica (tomará los valores `.TRUE` o `.FALSE`), se utiliza la cláusula L_n , n es la longitud de la variable. En lectura se asigna valor cierto (`.TRUE.`) a la variable, si al recorrer el dato de derecha a izquierda se encuentra una `T` antes que una `F`. Si no se encuentra ninguna de las dos, no se da valor. En escritura se imprime una `T` o una `F`, `.TRUE.` o `.FALSE.`, respectivamente.

Si se va a trabajar con una variable alfanumérica, debe especificarse A_n . Si n es mayor que la longitud del campo, rellena con blancos a la derecha. Si es menor, truncará.

```

C PROGRAMA EJEMPLO DE LECTURA Y ESCRITURA.
      INTEGER CAMPO
      REAL   LIMITE
      WRITE ( 0, 100 )
100   FORMAT ( 2X, 'VOY A LEER UNA VARIABLE ENTERA DE 4 POSICIONES' )
      READ ( 0, 200 ) CAMPO
200   FORMAT ( I4 )
      WRITE ( 0, 300 ) CAMPO, CAMPO
300   FORMAT ( 2X, 'LA ESCRIBO CON 3 ', I3, ', 1X, 'LA ESCRIBO CON 5 ', I5 )
      WRITE ( 0, 400 )
400   FORMAT ( 2X, 'LEO UNA VARIABLE REAL' )
      READ ( 0, 500 ) LIMITE
500   FORMAT ( F5.2 )
      WRITE ( 0, 600 ) LIMITE
600   FORMAT ( 2X, 'TIENE 2 POSICIONES ENTERAS ', F5.2 )
      STOP
      END
  
```

