BOTE BITS / McGraw-Hill

TRAVESURAS

con el ZX81

22 programas de Juegos

Graham Charlton • Mark Harrison • Dilwyn Jones

TRAVESURAS CON EL ZX81

(22 programas de juegos)

TRAVESURAS CON EL ZX81

22 Programas de juegos

Graham Charlton • Mark Harrison • Dilwyn Jones

TRADUCCION

Alfonso Camaño Liceras
Licenciado en Ciencias Exactas

REVISION TECNICA

Antonio Vaquero Sánchez
Catedrático de Informática
Facultad de Ciencias Físicas
Universidad Complutense de Madrid

McGraw-Hill

MADRID • BOGOTÁ • BUENOS AIRES • GUATEMALA • LISBOA • MÉXICO
NUEVA YORK • PANAMÁ • SAN JUAN • SANTIAGO • SAO PAULO
AUCKLAND • HAMBURGO • JOHANNESBURGO • LONDRES • MONTREAL
NUEVA DELHI • PARÍS • SAN FRANCISCO • SINGAPUR
ST. LOUIS • SIDNEY • TOKIO • TORONTO

TRAVESURAS CON EL ZX81. (22 programas de juegos)

ISBN: 968-451-705-X

Traducido de la primera edición en inglés de

BOGGLERS: 22 SMART GAMES PROGRAMS (2K TO 16K)
IN TIMEX/SINCLAIR BASIC

ISBN: 0-07-023959-2

Edición exclusiva para Ediciones La Colina, S. A. (España).

ISBN: 84-7615-001
Depósito legal: M. 26.372-1984

PRINTED IN SPAIN-IMPRESO EN ESPAÑA

Impreso en Lavel. Los Llanos, nave 6. Humanes (Madrid)

Contenido

Cómo usar este libro vii

Primera Parte: JUEGOS

1. Backgammon/Chaquete 3
2. Quartermass 16
3. Damas 20
4. Caza del Submarino 25
5. Gomoku/Go-Bang/Batuolos 30
6. Biorritmos 34
7. Eliza 40
8. Cuatro en raya 57
9. Othello/Reversi 62
10. El zorro y las ocas 67
11. Parejas 70
12. Cuatro campos 75
13. El surfista 79
14. El alquerque 82
15. El laberinto 87
16. Muertos vivientes 90
17. Poker de dados 93
18. Antimind 96
19. Nim 99

20. Cien 102
21. Tic-Tac-Toe 105
22. Solitario 114

Segunda parte: PROGRAMAS Y APLICACIONES DE UTILIDAD

23. Agenda de direcciones 121
24. Resistencias eléctricas y colores 125
25. Resultados del examen 133

APENDICES

1. Para acelerar la ejecución de programas 141
2. Caracteres expandidos 145
3. Variables del sistema útiles 155
4. Convertidor de decimal a hexadecimal 158
5. Convertidor de hexadecimal a decimal 160
6. Símbolos gráficos especiales 162

Cómo usar este libro

Programar con el BASIC de Timex/Sinclair sirve para entretener y educar a toda la familia. Los juegos y programas de utilidad de este libro están preparados para "correr" en una computadora Timex/Sinclair con 16K de memoria de acceso al azar (RAM). Todos los programas han sido comprobados previamente y una vez que hayan sido introducidos en la computadora le proporcionarán horas de entretenimiento. Tanto si reta a la computadora a una partida de backgammon, discute sus problemas con la psiquiatra Eliza o simplemente confecciona su agenda de direcciones, en todos los casos encontrará al Timex/Sinclair como un inteligente aliado, oponente o asistente.

Incluso aunque tenga experiencia con las computadoras Timex/Sinclair debe dedicar unos minutos a leer las siguientes líneas. Las computadoras Timex/Sinclair son máquinas poderosas y a la vez económicas, para usarlas debe estar familiarizado con el BASIC que es probablemente el lenguaje de computadoras más universalmente extendido y a la vez tiene que conocer las peculiaridades del teclado T/S, que está organizado de modo que el teclear sea una labor lo más eficiente y cómoda posible.

Una breve lectura de estas líneas junto con el *Manual del Usuario del Timex* que viene con su computadora, le permitirán conectar la computadora e introducir el primer programa en un tiempo aproximado de una hora.

Conexión del sistema

El *Manual del Usuario de Timex* contiene instrucciones claras y directas para conectar su computadora. Nos ha parecido que unos ajustes adicionales pueden hacer que el sistema sea más fácil de usar.

1. *El cable para conectar su computadora a la TV.* La mayor parte de las computadoras T/S producen una imagen de TV mediante caracteres negros sobre fondo blanco, por este motivo la pantalla aparece bastante brillante. Con el cable de 4 pies que viene con la computadora para conectarla con el televisor, se tiene que trabajar muy cerca de la pantalla y después de unas pocas horas de estar en esta posición se aprecia cansancio de ojos y un cierto dolor de cabeza. Recomendamos sustituir dicho cable por otro un poco más largo (6 u 8 pies). La mayor parte de los almacenes de componentes electrónicos tienen este tipo de cables y cuestan unos pocos cientos de pesetas. Con un cable un poco más largo podrá poner el televisor en el otro extremo de la mesa donde será perfectamente visible y producirá menos cansancio de ojos.
2. *El mando de los canales debajo de la computadora.* Algunas veces después de programar o de practicar un juego con la computadora, hemos desplazado la misma unos centímetros para permitir que otros tomasen el control. En algunas ocasiones al hacer esto se ha perdido por completo la imagen de la televisión. Después de investigar descubrimos la causa de este trastorno: el mando de selección de los canales que está debajo de la computadora debería estar incrustrado en la misma; sin embargo, en algunas de las computadoras con las que hemos trabajado no está suficientemente incrustrado. Como consecuencia al desplazar la computadora sobre la mesa puede rozarse el selector de canales pasando de uno a otro y se produce una pérdida momentánea de la imagen.

No tiene importancia, basta volver a seleccionar el canal adecuado para recuperar la imagen que no se ha perdido puesto que estaba almacenada en memoria. Si este problema se manifiesta insistentemente puede bloquear el selector de canales con un trozo de papel adherente.

También hemos observado que los consejos del fabricante sobre cuál de los dos canales (2 ó 3) debe usarse, no son siempre ciertos. De las tres máquinas que hemos utilizado, dos funcionaban mejor en el canal 3 y la tercera lo hacía en el canal 2. Puesto que las pruebas se hicieron en Nueva York donde el canal 2 está ocupado por la emisora CBS-TV, se explica que funcionara mejor el canal 3, pero esto no sucede siempre. No tenemos una explicación convincente de este hecho, pero preferimos prevenirle.

3. *Conexión del magnetófono.* Al cargar y guardar programas, se ha observado que tal y como se indica en el *Manual del Usuario de Timex* es mejor tener el volumen del magnetófono cerca de su punto más alto. De forma similar, se obtienen los mejores resultados al cargar y guardar programas, si el tono se coloca en el punto más agudo. Más aún, si su magnetófono tiene las dos posibilidades "mono" y "stereo", utilice la primera de ellas. En el caso de que únicamente sea "stereo" gire el balance completamente hacia la derecha o hacia la izquierda y déjelo en esa posición tanto para cargar como para guardar.

   El *Manual del Usuario Timex* contiene una sugerencia interesante sobre la forma de almacenar programas para que sean fáciles de encontrar: véase la página 14 del manual donde se explica cómo insertar una voz que anuncie el comienzo de cada programa que haya grabado en la cinta.

   Tres cosas más: Primero, es preferible pagar más por una cinta de buena calidad. Segundo, es aconsejable hacer una copia de seguridad de cada programa, preferiblemente en un cassette distinto. Tercero, debe grabar sus programas sólo en una cara del

cassette, puesto que sino las señales de una cara pueden interferir las señales de la otra.

4. Probablemente el mayor inconveniente de la computadora Timex/Sinclair es que el teclado es muy pequeño. Para obtener la mayor comodidad en la manipulación del teclado manténgale iluminado y como ya se ha mencionado a cierta distancia del televisor. Con unos pocos minutos de práctica comprobará que el teclado es de hecho muy manejable.

Teclado

Si no tiene experiencia con el teclado T/S, lea esta sección cuidadosamente y con el mismo cerca para comprobar lo que lee.

1. Cuando desee pulsar el número 1, asegúrese de que efectivamente presiona la tecla correspondiente al número 1 y no la correspondiente a la letra L. La tecla del 1 esta situada en la fila superior del teclado y en su extremo izquierdo.
2. No confunda el número Ø con la letra O. En los listados de programas de este libro así como en la imagen de su TV, apreciará que el número cero se imprime con una barra que lo cruza: Ø.
3. La letra I y el número 1 pueden parecer también similares en los listados de los programas, pero debe asegurarse de que los distingue. La letra I se representa en este libro así |, mientras que el número 1 se representa mediante 1.
4. En algunos programas se hace uso del subrayado (—) para indicar que debe dejar un espacio (presionando para ello la tecla SPACE). No intente pulsar la tecla del subrayado puesto que no la encontrará en el teclado de la computadora T/S. Sólo se utiliza dentro de comillas en los listados de programas, puesto que tan solo dentro de comillas puede ser crucial insertar el apropiado número de espacios.

5. Todas las palabras impresas en letras negrillas en los listados de los programas son palabras reservadas. Cada una de estas palabras es accesible mediante una única tecla. Si se intenta teclear letra a letra una de estas palabras, la computadora no la aceptará o el programa no se ejecutará correctamente. Por tanto siempre que encuentre una palabra en negrilla asegúrese de que accede mediante una única tecla.

6. Los siguientes símbolos matemáticos son también accesibles mediante una sola pulsación
   - < > que significa "distinto" y se encuentra en la tecla T.
   - > = que significa "mayor o igual" y se encuentra en la tecla Y.
   - < = que significa "menor o igual" y se encuentra en la tecla R.

7. Recuerde que cada tecla del T/S puede tener hasta 5 significados diferentes dependiendo del modo en el que se esté operando. El modo se indica mediante el cursor del siguiente modo:
   - **K** Modo palabra clave ("keyword"). Cuando se está en este modo, se accede a la palabra impresa en blanco sobre la tecla.
   - **L** Modo literal. Cuando se está en este modo se accede al carácter en negro oscuro que esté sobre la tecla.
   - **F** Modo función. En este modo se accede a la palabra impresa en blanco debajo de la tecla.
   - **G** Modo gráfico. Si se pulsa una tecla en este modo, el carácter en negro oscuro de la tecla aparece en "negativo". (Esto significa que el carácter aparecerá en blanco en la pantalla, sobre un pequeño cuadrado negro.) Si se mantiene pulsada la tecla SHIFT mientras que se está en el modo gráfico, algunas de las teclas le darán acceso a unos pequeños símbolos gráficos que se muestran en la esquina inferior derecha de cada tecla.

Almacenamiento de programas en cinta

Después de haber tecleado un programa, comprobado que funciona y decidido conservarlo, puede guardarlo grabándole en cinta magnética, de la misma forma que se graba una pieza musical. Todo lo que tiene que hacer es pulsar **SAVE** seguida del nombre elegido para su programa entre comillas. Por ejemplo, si decide llamar a su programa CAZA DEL SUBMARINO, deberá escribir en su computadora:

SAVE CAZA DEL SUBMARINO

Ahora debe colocar su magnetófono en posición de grabar. Después pulse la tecla ENTER y el programa (o más exactamente una copia del programa, puesto que el mismo continuará en la memoria principal de la computadora) será grabado en la cinta. Sabrá cuando se ha concluido la grabación por medio del código Ø/Ø que aparecerá en la esquina inferior izquierda de la pantalla. En ese momento puede desconectar el magnetófono.

Carga de un programa

Para cargar un programa desde la cinta magnética a la computadora, pulse **NEW** seguido de ENTER. A continuación puede escribir:

LOAD CAZA DEL SUBMARINO

que busca y carga el programa llamado CAZA DEL SUBMARINO, o bien:

LOAD " "

que busca y carga el primer programa completo que encuentre en la cinta.

Recordatorios importantes

1. Anótese el nombre elegido para el programa que esté grabando, incluyendo los espacios entre las palabras que puedan existir. Si guarda un programa con

el nombre CAZA DEL SUBMARINO y después intenta cargarlo llamándole mediante CAZADELSUBMARINO, la computadora no le encontrará.

2. Tanto guardando como cargando es preferible que el magnetófono esté en posición de "mono" (en lugar de "stereo"), de esta forma funciona mejor. Si su magnetófono es exclusivamente "stereo", debe situar el control del balance totalmente a la derecha o a la izquierda. Además el volumen de grabación debe estar bastante alto, tres cuartas partes del total, el tono debe estar en su punto más agudo.

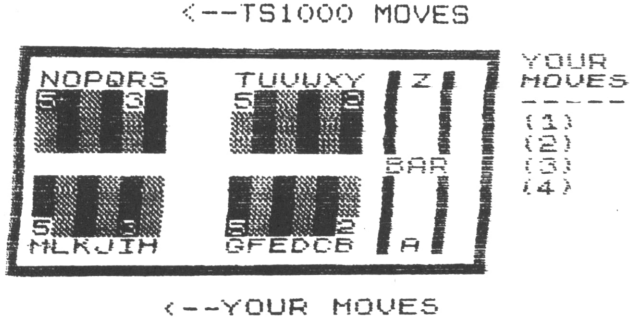
Primera
parte

JUEGOS

1

Backgammon/Chaquete

Este programa permite a la computadora retarle a jugar al Backgammon. El programa insiste en que los movimientos deben ser legales rechazando cualquier intento de engaño.



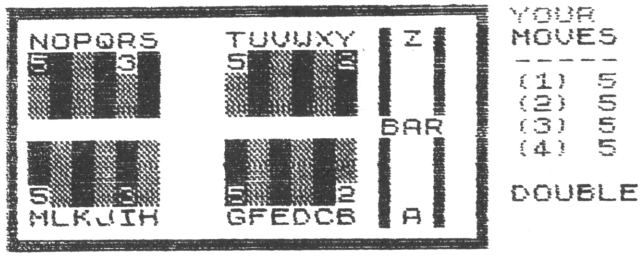
El tablero que aparece en pantalla está limitado por las posibilidades gráficas de la computadora. Por ejemplo, lo que normalmente son puntos triangulares, aquí se representan mediante rectángulos. Las fichas individuales no aparecen, en su lugar se muestra el número de fichas en cada lugar con un dígito entre ∅ y 9, o si hubiera más piezas se utilizan las letras entre A y Z para representar los números entre 10 y 15 del siguiente modo:

A = 10; B = 11; C = 12; D = 13; E = 14; F = 15.

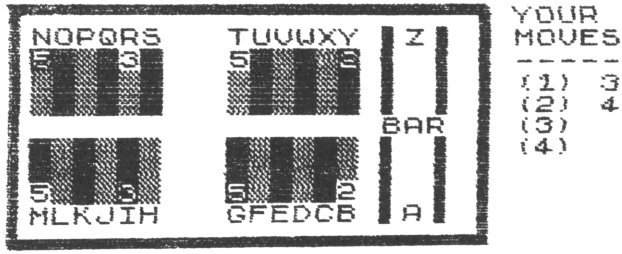
Esta es la notación hexadecimal clásica y se acostumbrará a ella rápidamente. De todas formas, sólo ocasionalmente se tendrán más de 9 fichas en un lugar.

Las fichas de la computadora aparecen como caracteres "en negativo" (caracteres blancos sobre fondo negro) y se mueven desde el punto Y hacia el B. Las fichas rescatadas por la computadora van a parar a A. Las fichas de la computadora que son capturadas retornan a Z. El oponente mueve sus video-fichas desde B hacia Y, rescatando sus fichas, hacia Z y las fichas que le capturan retornan a A.

El programa utiliza su propio dado y los movimientos se registran a la derecha del tablero. Normalmente aparecen dos números representando los dos lanzamientos del dado. Si se lanza un "doble" aparecen cuatro números, junto con un mensaje que le recuerda que el lanzamiento ha sido un "doble".



Si en algún momento no puede mover (por ejemplo, si tiene una ficha en la barrera que no puede usarse) entonces pulse simplemente ENTER cuando la computadora solicite su movimiento. La computadora comprobará que Vd. no puede mover (en caso de error o engaño) y realizará su movimiento o le dirá que mueva legalmente si es posible.



Si en algún momento desea obtener el programa, pulse BREAK si es el turno de la computadora o **STOP** (shift-A) en caso de que sea su turno. El programa detecta automáticamente cuando ha ganado o perdido el juego. Para introducir su movimiento, precisa introducir primero el lanzamiento del dado elegido así como el lugar desde el cual se va a mover, esto es, una letra y un número. Letra y número pueden ser introducidas en cualquier orden, puesto que la computadora los clasificará. Por ejemplo, si ha obtenido un tres con el dado y desea mover una de sus piezas desde el punto B, puede introducir "B3" o "3B". Si el movimiento es legal observará cómo se mueven las fichas en el tablero. En otro caso tendrá que reintroducir su movimiento.

Programa 1: Backgammon

```
  1Ø REM BACKGAMMON
  2Ø REM RUNS IN 6K
  3Ø REM INITIALIZE
  5Ø GOSUB 9ØØØ
  6Ø REM BY DILWYN JONES 1982
 999 REM PLAYER MOVES
1ØØØ PRINT AT Ø,23;"YOUR_ _ _"†
1ØØ5 LET B$="1"
1ØØ9 REM ROLL DICE
1Ø1Ø GOSUB 2ØØØ
1Ø3Ø DIM M$(2)
1Ø4Ø INPUT M$
1Ø42 RAND
1Ø45 IF M$(1)="_STOP_" THEN STOP
1Ø5Ø IF M$="_ _" THEN GOTO 12ØØ
```

† Las notas entre comillas indican el número de espacios que se deben dejar.

```
1Ø55  IF M$(1)>="1" AND
      M$(1)<="6" AND M$(2)>="A" AND M$(2)
      <="Y" THEN LET M$=M$(2)+M$(1)
1Ø6Ø  IF M$(1)<"A" OR M$(1)>"Y" OR M$(2)<"1"
      OR M$(2)>"6" THEN GOTO 1Ø4Ø
1Ø7Ø  FOR G=1 TO D
1Ø8Ø  IF D$(G)=M$(2) THEN GOTO 111Ø
1Ø9Ø  NEXT G
11ØØ  GOTO 1Ø4Ø
111Ø  LET FROM=CODE M$-37
112Ø  LET TO=FROM+VAL M$(2)
1125  IF FROM<>1 AND A$(2,1)<>A$(1,1) THEN
      GOTO 1Ø4Ø
113Ø  IF A$(2,FROM)<"1" OR A$(2,FROM)>"F" THEN
      GOTO 1Ø4Ø
1135  IF TO<26 THEN IF A$(2,TO)="1" THEN GOTO
      3ØØØ
114Ø  IF TO <26 THEN IF A$(2,TO)>="2" AND
      A$(2,TO)<="F" THEN GOTO 1Ø4Ø
115Ø  IF TO<26 THEN GOTO 4900
1155  FOR H=1 TO 19
116Ø  IF A$(2,H)>="1" AND A$(2,H)<="F" THEN
      GOTO 1Ø4Ø
1165  NEXT H
118Ø  GOTO 4ØØØ
1198  REM PLAYER NULL INPUT
1199  REM CAN PLAYER MOVE?
12ØØ  LET FLAG=Ø
```

```
12Ø5  FOR H=1 TO 25
121Ø  IF A$(2,H)<"1" OR A$(2,H)>"F" THEN GOTO
      127Ø
122Ø  FOR G=1 TO D
123Ø  IF D$(G)="_" THEN GOTO 126Ø
124Ø  IF H+VAL D$(G)<26 THEN IF A$(2,H+VAL
      D$(G))<"2" OR A$(2,H+VAL D$(G))>"F"
      THEN GOTO 13ØØ
125Ø  IF H+VAL D$(G)>=26 AND NOT FLAG THEN
      GOTO 13ØØ
126Ø  NEXT G
1265  IF A$(2,1)<>A$(1,1) THEN GOTO 15ØØ
1267  IF H<=19 THEN LET FLAG=1
127Ø  NEXT H
128Ø  GOTO 15ØØ
13ØØ  PRINT AT 14,Ø; "YOU_CAN_MAKE_A_
      LEGAL_MOVE"
131Ø  INPUT M$
132Ø  PRINT AT 14,0;"_ _ _ _ _ _ _ _ _ _ _ _ _ _
      _ _ _ _ _ _ _ _ _ _ _ _ _ _"
133Ø  GOTO 1Ø45
1499  REM TS1000_MOVE
15ØØ  PRINT AT Ø,23; "TS1ØØØ"
15Ø5  LET B$="1"
15Ø9  REM ROLL DICE
151Ø  GOSUB 2ØØØ
152Ø  LET FLAG=Ø
1525  LET BEAROFF=Ø
```

```
1530  FOR F=26 TO 2 STEP -1
1540  LET FROM=F
1545  IF FROM<=7 AND FLAG=0 AND NOT
      BEAROFF THEN GOTO 1700
1550  IF A$(2,FROM)<"1" OR A$(2,FROM)>"F"
      THEN GOTO 1650
1560  FOR G=1 TO D
1570  IF D$(G)="_" THEN GOTO 1620
1580  LET TO=FROM-VAL D$(G)
1590  IF TO<2 AND FLAG=0 THEN GOTO 4000
1600  IF TO>1 THEN IF A$(2,TO)="1" THEN GOTO
      3000
1610  IF A$(2,TO)<"1" OR A$(2,TO)>"F" THEN GOTO
      4900
1620  NEXT G
1630  IF A$(2,26)<>A$(1,26) THEN GOTO 1000
1640  IF FLAG=0 THEN LET FLAG=1
1650  NEXT F
1660  GOTO 1000
1699  REM TS1000 BEARING OFF
1700  FOR H=7 TO 2 STEP -1
1710  IF A$(2,H)=A$(1,H) THEN GOTO 1780
1720  LET FROM=H
1730  FOR G=1 TO D
1740  IF D$(G)="_" THEN GOTO 1770
1750  LET TO=FROM-VAL D$(G)
1760  IF TO<2 THEN GOTO 4000
1770  NEXT G
```

```
1780  NEXT H
1785  LET F=7
1787  LET BEAROFF=1
1790  GOTO 1540
1999  REM DICE SUBROUTINE
2000  DIM D$(4)
2005  PRINT AT 8,23;"_ _ _ _ _ _"
2010  LET D$(1)=STR$ (INT (RND*6)+1)
2020  LET D$(2)=STR$ (INT (RND*6)+1)
2030  IF D$(1)=D$(2) THEN LET D$(3 TO )=D$(  TO 2)
2040  LET D=2+(2 AND D$(1)=D$(2))
2050  PRINT AT 3,27;D$(1);TAB 27;D$(2);TAB
      27;D$(3);TAB 27;D$(4)
2070  IF D=2 THEN RETURN
2079  REM DOUBLE THROW
2080  FOR F=1 TO 20
2090  PRINT AT 8,22;"DOUBLE";AT 8,22; "_
      DOUBLE_"
2100  NEXT F
2110  RETURN
3000  REM HIT A BLOT
3010  LET A$(2,TO)=B$
3020  LET A$(2,FROM)=(A$(1,FROM) AND
      A$(2,FROM)=B$)+(CHR$ (CODE A$(2,FROM)-1)
      AND A$(2,FROM)>B$)
3030  IF B$="1" THEN LET A$(2,26)=("1" AND
      A$(2,26)=A$(1,26))+(CHR$ (CODE A$(2,26)+1)
      AND A$(2,26)>="1")
```

```
3Ø4Ø  IF B$="1" THEN LET A$(2,1)=("1" AND
      A$(2,1)=A$(1,1))+(CHR$(CODE A$(2,1)+1) AND
      A$(2,1)>="1")
3Ø5Ø  GOTO 5ØØØ
3999  REM BEARING OFF
4ØØØ  IF B$="1" THEN LET P$=CHR$(CODE P$+1)
4Ø1Ø  IF B$="1" THEN LET Z$=CHR$(CODE Z$+1)
4Ø2Ø  LET A$(2,FROM)=(A$(1,FROM) AND
      A$(2,FROM)=B$)+(CHR$ (CODE A$(2,FROM)-1)
      AND A$(2,FROM)>B$)
4Ø4Ø  IF B$="1" AND P$>"Ø" THEN PRINT AT
      3,18;P$
4Ø45  IF B$="1" AND Z$>"Ø" THEN PRINT AT
      7,18;Z$
4Ø5Ø  GOTO 5ØØØ
4899  REM SET FROM/TO POINTS
49ØØ  IF TO>1 AND TO<26 THEN LET A$(2,TO)=(B$
      AND A$(2,TO)=A$(1,TO))+(CHR$ (CODE
      A$(2,TO)+1) AND A$(2,TO)>=B$)
491Ø  LET A$(2,FROM)=(A$(1,FROM) AND
      A$(2,FROM)=B$)+(CHR$ (CODE A$(2,FROM)-1)
      AND A$(2,FROM)>B$)
5ØØØ  REM PRINT MOVES, DICE, WON?
5Ø1Ø  LET D$(G)="_"
5Ø2Ø  PRINT AT 2+G,27;D$(G)
5Ø3Ø  PRINT AT CODE Y$(FROM),CODE
      X$(FROM);A$(2,FROM)
5Ø4Ø  IF TO>1 AND TO<26 THEN PRINT AT CODE
      Y$(TO),CODE X$(TO);A$(2,TO)
```

```
5050 PRINT AT CODE Y$(1), CODE X$(1);A$(2,1);
     AT CODE Y$(26), CODE X$(26); A$(2,26)
5060 IF P$="F" OR Z$="F" THEN GOTO (6000
     AND P$="F")+(6500 AND Z$="F")
5070 IF D$="_ _ _ _" THEN GOTO (1000 AND B$=
     "1")+(1500 AND B$="1")
5080 GOTO (1030 AND B$="1")+(1540 AND B$="1")
6000 REM PLAYER HAS WON
6010 FOR F=1 TO 50
6020 PRINT AT 16,10;"YOU WIN"; AT 16,10;" YOU_
     WIN "
6030 NEXT F
6035 IF RND<.3 THEN PRINT AT 18,8; "YOU_
     WERE_LUCKY"
6040 STOP
6500 REM TS1000 HAS WON
6510 FOR F=1 TO 50
6520 PRINT AT 16,11; "I WIN";AT 16,11;" I_WIN "
6530 NEXT F
6535 IF RND<.3 THEN PRINT AT 18,8; "TS1000_
     RULES, OK?"
6540 STOP
8999 REM INITIALIZE
9000 CLS
9001 PRINT AT $4,10;"■■■■■■■■■■
     ■■■";TAB 10; "■BACKGAMMON■";TAB
     10; "■■■■■■■■■■■■■■
9002 PRINT AT 10,15; "BY"; AT 12, 10; "DILWYN_
     JONES"
```

```
9005  FOR F=1 TO 70
9006  NEXT F
9008  CLS
9010  PRINT "
      "
9020  PRINT "NOPQRS _ _ _ TUVWXY _Z_
      _"
9030  PRINT "5 3 _ _ _5
      2_ _ _ _"
9040  PRINT "
      _ _ _
      _ _ _ _"
9050  PRINT "
      _ _ _
      _ _ _"
9060  PRINT "_ _ _ _ _ _ _ _ _ _ _ _ _ _ _
      _BAR_"
9070  PRINT "
      _ _ _
      _ _ _"
9080  PRINT "
      _ _ _
      _ _ _"
9090  PRINT "5 3 _ _ _5 2
      _ _ _"
9100  PRINT "MLKJIH_ _ _GFEDCB_A _"
9110  PRINT "
      "
9115  DIM A$(2,26)
9120  LET A$(1)="_
      _"
9130  LET A$(2)="_2 5 3 5 5
      3 5 2_"
```

```
9140 LET Y$="▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
9150 LET X$=">?:$£"+CHR$11+"▒▗▐▖▀▝▘▝▀▖▌▗▒"+CHR$ 11+"£$:?>"
9160 PRINT AT 1,23;"MOVES";TAB 23;"------";
     TAB 23;"(1)";TAB 23;"(2)";TAB 23;"(3)";TAB
     23;"(4)"
9170 LET P$="Ø"
9180 LET Z$="Ø"
9200 RETURN
9900 SAVE "BACKGAMMON"
9910 RUN
```

Notas al programa

El programa ha sido escrito para ejecutarse en el modo lento **(SLOW),** sin traducirlo al lenguaje máquina. La mayor parte de los movimientos se realizan en uno o dos segundos, en los casos más difíciles puede llegarse a los cinco segundos. Para que el programa se ejecute tan rápidamente ha sido preciso añadir toda clase de señaladores y otras facilidades. Esto amplía considerablemente la longitud del programa. Una gran parte de la memoria está ocupada por las rutinas que dibujan en la pantalla, las sentencias de captura de errores y las sentencias **REM.** Si se simplificara, los usuarios con sistemas de 4K podrían probablemente usar el programa, cuyo núcleo ocupa alrededor de 3K.

El programa está generosamente salpicado de sentencias **REM,** por lo que sólo se precisa una explicación breve.

A$ es la matriz principal. A$(1) contiene la situación del tablero sin piezas. A$(2) contiene la situación de las casillas del tablero comprendidas entre la A y la Z. Apréciese lo ajustado que le viene el alfabeto al Backgammon. Y$ contiene los caracteres cuyo **CODE** corresponde a la coordenada

Y de la casilla considerada y se usa en las rutinas de impresión. X$ es similar y contiene la coordenada X. P$ y Z$ contienen el número de piezas rescatadas por el oponente y la computadora respectivamente.

M$ es la cadena que contiene el movimiento del oponente. D$ es la cadena que contiene el lanzamiento del dado. FROM es la variable que se refiere al punto desde el cual se va a mover y TO (que se compone de las letras T y O no de la palabra reservada **TO** que se alcanza mediante shift-4) se refiere obviamente a la casilla a la cual se va a mover. Estas son las variables más importantes.

La parte del programa que comienza en la línea 1000 acepta el movimiento del oponente y decide si el movimiento es legal. A partir de la línea 1200 comienza un bloque del programa que escudriña el tablero al pulsar el oponente ENTER después de haber introducido su movimiento para comprobar si efectivamente puede realizarse algún movimiento. La línea 1500 es el principio de un bloque que determina el movimiento del computador. El bloque que empieza en la línea 1700 comprueba cuando alguna ficha de la computadora ha sido rescatada si las reglas del juego lo permiten (la computadora no tiene fichas en la barrera y todas las fichas de la computadora están en juego o en las casillas entre la G y la B).

En la línea 2000 comienza una subrutina que genera el lanzamiento del dado. Dicho lanzamiento genera un carácter entre "1" y "6". Normalmente hay sólo dos, pero si el lanzamiento fue un doble (ambos dados con la misma puntuación), D$ recibe cuatro lanzamientos. D es la varible que dice cuántos números hay en D$. Las líneas 2080, 2090 y 2100 imprimen intermitentemente la palabra DOUBLE en negativo y positivo alternativamente en la pantalla, para llamar la atención del oponente. La rutina de la línea 3000 toma las medidas apropiadas si se captura una ficha. B$ cumple dos funciones, por un lado informa a la computadora su movimiento y por otro informa qué carácter rellenará un punto si el movimiento se ha realizado a una casilla vacía.

El siguiente bloque maneja el rescate. Las comprobaciones apropiadas de si está permitido o no el rescate se han

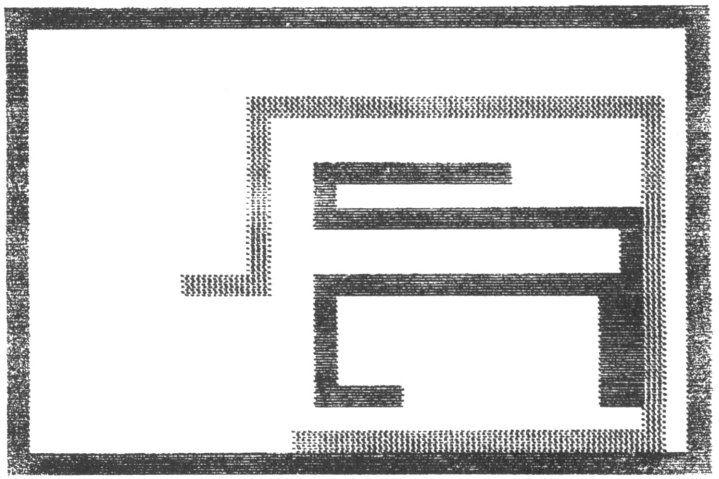
realizado con anterioridad. El bloque que empieza en la línea 4900 da a las variables FROM y TO valores apropiados. La línea 5000 marca el comienzo de una rutina que coloca el tablero y lo actualiza, imprimiendo además los lanzamientos del dado.

Entre las líneas 6000 y 6540 se encuentran las rutinas de la "victoria". Aquí es donde el programa termina si decide que uno de los jugadores ha ganado. La subrutina de la línea 9000 inicializa el programa, la pantalla, etc. Las líneas 9900 y 9910 posibilitan la ejecución automática. Una vez que se ha tecleado el programa guárdese pulsando **RUN** 9900. El programa se ejecutará entonces automáticamente cuando se cargue desde la cinta.

2

Quartermass

El objeto de este programa es rodear a su enemigo, quien a su vez intentará rodearle a Vd. Pulse "W" para mover arriba, "X" para mover abajo, "A" para mover a la derecha y "D" para mover a la izquierda. Su trazo es el discontinuo y el de la computadora es el negro.



Programa 2: Quartermass

```
25  SLOW
30  PRINT AT 0,4;"QUATERMASS.";AT 1,4;"■■■■
    ■■■■■■■■";AT 3,4;"THE_OBJECT_OF_
```

```
      THE_GAME_IS_TO_SURROUND_YOUR_
      ENEMY_WHO_IN_TURN_IS_ALSO_TRYING_
      TO_SURROUND_YOU."
35    PRINT AT 7,Ø;"PRESS:",,"""W""_TO_MOVE_
      UP",,"""X""_TO_MOVE_DOWN","""A""_TO_
      MOVE_RIGHT","""D""_TO_MOVE_LEFT"
4Ø    PRINT AT 13,Ø;"PLAYERS_COLOR_▩▩▩";
      AT 15,Ø;"ENEMYS COLOR_■■■"
45    PRINT AT 21,Ø;"PRESS_ENTER_TO_BEGIN."
5Ø    INPUT Q$
55    CLS
6Ø    RAND
65    LET A$="ØØ1Ø-1Ø33-33ØØ1Ø-1Ø33"
7Ø    FOR I=Ø TO 31
75    PRINT AT Ø,I;"■";AT 2Ø,I;"■";AT I AND I
      <2Ø,31;"■■"
8Ø    NEXT I
85    LET D=255*PEEK 16397+PEEK 16396
9Ø    LET X=D+4Ø5
95    LET Z=D+254
1ØØ   LET ZD=-1
1Ø5   LET XD=1
11Ø   POKE X,8
115   POKE Z,128
12Ø   IF INKEY$="" THEN GOTO 12Ø
125   LET D= (33 AND INKEY$="X")-(33 AND
      INKEY$="W") + (1 AND INKEY$="D")-(1 AND
      INKEY$="A")
13Ø   IF D<>Ø THEN LET XD=D
```

```
135  LET X=X+XD
140  IF PEEK X<>0 THEN GOTO 230
145  POKE X,8
150  IF PEEK (Z+ZD)+PEEK (Z+2*ZD)+PEEK
     (Z+3*ZD)=0 AND RND<0.9 THEN GOTO 215
155  LET D=3*INT (RND*4)+1
160  FOR I=3 TO 1 STEP -1
165  FOR J=D TO D+9 STEP 3
170  LET ZD=VAL A$(J TO J+2)
175  FOR K=1 TO I
180  IF PEEK (Z+K*ZD)<>0 THEN GOTO 195
185  NEXT K
190  GOTO 215
195  NEXT J
200  NEXT I
205  PRINT AT 21,0;"YOU_WIN"
210  GOTO 235
215  LET Z=Z+ZD
220  POKE Z,128
225  GOTO 125
230  PRINT AT 21,0;"TS1000_WIN"
235  INPUT Q$
240  IF Q$="" THEN GOTO 55
245  STOP
```

Ejemplo de ejecución

QUATERMASS

Nota del traductor

La traducción al castellano del texto que aparece al ejecutar el programa es la siguiente:

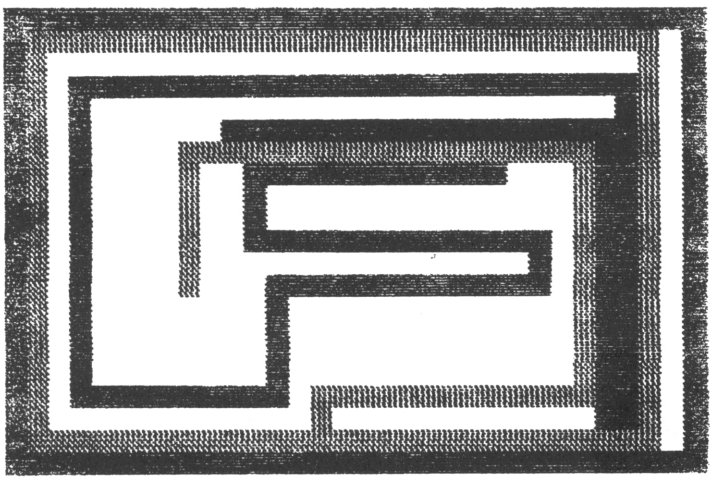
EL OBJETO DEL JUEGO ES CERCAR
AL ENEMIGO, QUIEN A SU VEZ INTENTARA
CERCARLE A USTED.

PULSE
"W" PARA MOVER HACIA ARRIBA
"X" PARA MOVER HACIA ABAJO
"A" PARA MOVER HACIA LA DERECHA
"D" PARA MOVER HACIA LA IZQUIERDA

COLOR DEL JUGADOR ▒▒▒

COLOR DEL ENEMIGO ■■■

PULSE ENTER PARA EMPEZAR



TS1000 WIN

3

Damas

Este programa construye un juego de damas sorprendentemente bueno. Tendrá que concentrarse para derrotar a la computadora.

La cadena de la línea 4000 imprime el tablero, la línea 4010 lee un extraño conjunto de caracteres de cuyos códigos consigue información sobre la posición de cada ficha en la cadena A$. Estos códigos se introducen en un vector en las líneas comprendidas entre la 4030 y la 4050, de este modo la computadora conoce en todo momento dónde están sus piezas. La línea 4060 contiene información que utiliza la computadora junto con el contenido de la línea 1040 y líneas similares para buscar los cuadros a los que se puede desplazar diagonalmente.

Las líneas comprendidas entre 1000 y 2990 contienen el algoritmo que permite a la computadora tomar decisiones. Entre las líneas 1010 y 1170 se mira cada ficha para comprobar si puede "comer". Si es posible la captura la realiza. Esta parte del algoritmo es razonablemente lenta al principio del juego cuando la computadora tiene un montón de fichas en el tablero, pero hacia el final del juego la computadora mueve casi instantáneamente. Las capturas múltiples se controlan también desde esta rutina (de forma muy sencilla como se verá) aunque se requerirá un análisis detallado para entenderlo. La línea 1120 comprueba si alguna pieza se ha convertido en dama, en cuyo caso, manda el control a la línea 2990 donde se realiza la coronación.

Las líneas comprendidas entre la 2000 y la 2130 son para un movimiento que no implique captura. La computadora investiga primero si puede mover una ficha donde no esté en peligro de captura y si no encuentra ninguno busca un movimiento legal aunque coloque una de sus fichas en peligro. Si no encuentra ningún movimiento legal, mediante la línea 2140 imprime el mensaje YOU WIN (Vd. gana).

Las líneas desde la 3000 a la 3230 aceptan el movimiento del oponente. El movimiento se introduce en dos cadenas, G$ y H$ cada una de ellas contiene una letra entre la A y la H y un número del 1 al 8. La primera cadena representa la casilla desde la cual se mueve y la segunda la de destino. Después las cadenas aparecen en pantalla para que compruebe si el movimiento es el que Vd. quería realizar, pulse "N" si ha habido error o cualquier otra tecla si es correcto.

Si captura una pieza, se reimprimirá el tablero y la computadora le preguntará si desea capturar de nuevo con la misma ficha. Si puede y desea pulse "Y".

Programa 3: Damas

```
  1Ø  GOTO 4ØØØ
 1ØØ  PRINT AT 6,11;
 11Ø  FOR A=1 TO 91 STEP 1Ø
 12Ø  PRINT TAB 11;A$(A TO A+9)
 13Ø  NEXT A
 14Ø  RETURN
1ØØØ  GOSUB 1ØØ
1Ø1Ø  LET Q=Ø
1Ø2Ø  FOR B=1 TO 12
1Ø22  LET C=B(B)
1Ø25  IF A$(C)<>"X" AND A$(C)<>"$" THEN GOTO
      117Ø
1Ø3Ø  FOR X=1 TO 4
1Ø4Ø  LET N=CODE B$(X)-5Ø
```

```
1Ø5Ø  IF NOT ((X<3 AND A$(C)="X") OR A$(C)=
      "$") THEN GOTO 115Ø
1Ø6Ø  IF NOT (A$(C+N)="O" OR A$(C+N)="£")
      THEN GOTO 115Ø
1Ø7Ø  IF NOT (A$(C+2*N)="_" THEN GOTO 115Ø
1Ø8Ø  LET A$(C+2*N)=A$(C)
1Ø9Ø  LET A$(C)="_"
11ØØ  LET A$(C+N)="_"
11Ø5  LET B(B)=C+2*N
111Ø  LET C=C+2*N
112Ø  IF C>8Ø THEN GOTO 299Ø
1125  GOSUB 1ØØ
113Ø  LET Q=1
114Ø  GOTO 1Ø3Ø
115Ø  NEXT X
116Ø  IF Q=1 THEN GOTO 3Ø1Ø
117Ø  NEXT B
2ØØØ  FOR A=1 TO 2
2Ø1Ø  FOR B=1 TO 12
2Ø15  LET C=B(B)
2Ø17  IF A$(C)<>"X" AND A$(C)<>
      "$" THEN GOTO 212Ø
2Ø2Ø  FOR X=1 TO 4
2Ø3Ø  LET N=CODE B$(X)-5Ø
2Ø4Ø  IF NOT ((X<3 AND A$(C)="X") OR A$(C)="$")
      THEN GOTO 212Ø
2Ø5Ø  IF NOT (A$(C+N)="_") THEN GOTO 211Ø
2Ø6Ø  IF A=1 AND ((A$(C+2*N)="O" OR
      A$(C+2*N)="£") OR ((A$(C+N+2Ø-ABS
```

```
      N)="O" OR A$(C+N+20-ABS N)="£") AND
      A$(C+N-20+ABS N)="_")) THEN GOTO
      2110
2070  LET A$(C+N)=A$(C)
2080  LET A$(C)="_"
2085  LET B(B)=C+N
2090  IF C+N>80 THEN LET A$(C+N)="$"
2100  GOTO 3000
2110  NEXT X
2120  NEXT B
2130  NEXT A
2140  PRINT AT 17,12;"YOU WIN";Z
2990  LET A$(C)="$"
3000  GOSUB 100
3010  PRINT AT 17,8;"FROM_?_"
3020  INPUT G$
3030  PRINT AT 17,13;G$;"_TO_?_"
3040  INPUT H$
3050  PRINT AT 17,19;H$;"_?"
3060  INPUT Z$
3080  PRINT AT 17,8;"_ _ _ _ _ _ _ _ _ _ _ _ _ _
      _ _"
3090  IF Z$="N" THEN GOTO 3000
3100  LET G=10* (CODE G$(1)-37)+CODE G$(2)-27
3110  LET H=10*(CODE H$(1)-37)+CODE H$(2)-27
3120  LET A$(H)=A$(G)
3130  IF H<20 THEN LET A$(H)="£"
3140  LET A$(G)="_"
```

```
3150  IF ABS (H-G)<>22 AND ABS (H-G)<>18
      THEN GOTO 1000
3160  LET A$((H+G)/2)="_"
3170  GOSUB 100
3180  PRINT AT 17,6; "CAN_YOU_JUMP_AGAIN?"
3190  LET Z$=INKEY$
3200  IF Z$="" THEN GOTO 3190
3210  PRINT AT 17,6;"_ _ _ _ _ _ _ _ _ _ _ _ _
      _ _ _ _ _ _ _"
3220  IF Z$="Y" THEN GOTO 3010
3230  GOTO 1010
4000  LET A$="▒12345678▒A■X■X■X■XABX
      ■X■X■X■BC■X■X■X■XCD_■_■_
      ■_■DE■_■_■_■_EFO■O■O■O■
      FG■O■O■O■OGHO■O■O■O■H▒
      12345678▒"
4010  LET B$="$<?),-0/B759"
4020  DIM B(12)
4030  FOR B=1 TO 12
4040  LET B(B)=CODE B$(B)
4050  NEXT B
4060  LET B$="XVDB"
4070  PRINT AT 2,2;"DO_YOU_WANT_TO_GO_
      FIRST?_Y/N"
4080  LET Z$=INKEY$
4090  IF INKEY$="" THEN GOTO 4080
4100  PRINT AT 2,2;"CHECKERS_BY_GRAHAM_
      CHARLTON "
4110  IF Z$="Y" THEN GOTO 3000
4120  GOTO 1000
```

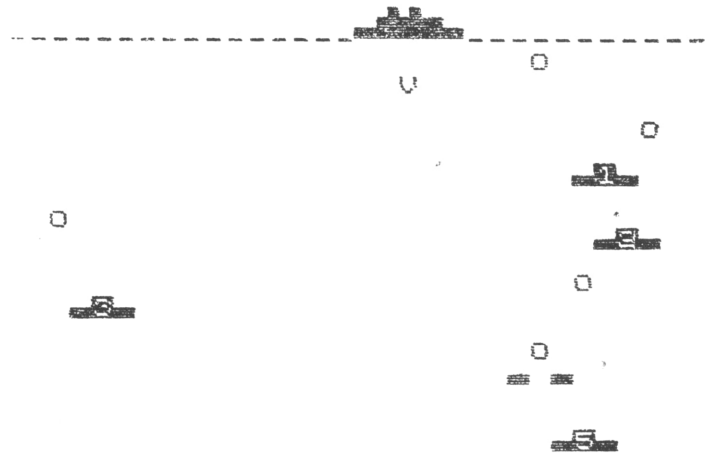
4

Caza del submarino

Es Vd. el capitán de un destructor y su objetivo es localizar y destruir cinco submarinos enemigos.

Puede lanzar una carga de profundidad cada vez. Cada submarino puede, a su vez, lanzar una bomba que flotará verticalmente hasta la superficie hundiendo cualquier buque que encuentra a su paso.

Sus controles son las teclas "M" y "N". Pulsando la tecla "B" se deja caer una carga de profundidad. Su tiempo aparecerá en la pantalla si sobrevive. En el juego se trata también de emplear el mínimo tiempo.



Progama 4: Caza del submarino

```
 5  GOSUB 500
10  LET L=1E9
15  CLS
20  DIM A(5)
25  DIM B(5,2)
30  FOR I=1 TO 5
35  LET A(I)=INT (RND*57-28)
40  LET B(I,1)=2+3*I
45  LET B(I,2)=ABS A(I)+2
50  NEXT I
55  LET Z=15
60  LET N=0
65  LET B=0
70  FOR I=0 TO 31
75  PRINT AT 1,I;"-"
80  NEXT I
85  POKE 16437,255
90  POKE 16436,255
95  FOR I=1 TO 5
100 PRINT AT 0,Z;"_ _▟▟▄_"; AT 1,Z;"-▄
    ▄▄▄▄_"
101 IF A(I)>98 THEN GOTO 115
102 PRINT AT 4+3*I,ABS A(I);"_▄"+CHR$ (156
    +I)+"▄_"
105 LET A(I)=A(I)+1
110 IF A(I)=29 THEN LET A(I)=-27
115 IF B(I,1)>=1 THEN GOTO 135
117 IF B(I,1)=-1 THEN GOTO 122
```

```
118 PRINT AT 1,B(I,2);"-"
120 LET B(I,1)=-1
122 IF A(I)>98 THEN GOTO 150
125 LET B(I,1)=2+3*I
130 LET B(I,2)=ABS A(I)+2
135 PRINT AT B(I,1)+1,B(I,2);"_";AT B(I,1),B(I,2);"O"
140 IF B(I,1)=1 AND B(I,2)-Z>0 AND B(I,2)-Z<6
    THEN GOTO 245
145 LET B(I,1)=B(I,1)-1
150 LET Z=Z+(INKEY$="M" AND
    Z<=24)-(INKEY$="N" AND Z>=1)
155 NEXT I
160 IF INKEY$<>"B" THEN GOTO 175
162 IF B<>0 THEN PRINT AT B-1,BB;"_"
165 LET BB=Z+3
170 LET B=3
175 IF B<=20 THEN GOTO 190
180 PRINT AT 20,BB;"_"
185 LET B=0
190 IF B=0 THEN GOTO 95
195 PRINT AT B-1,BB;"_";AT B,BB;
200 LET B=B+1
205 LET P=PEEK (256*PEEK 16399+PEEK 16398)
206 IF P=0 OR P=52 THEN GOTO 235
210 LET N=N+1
212 LET A((B-4)/3)=99
215 PRINT AT B-1,BB-2;"_ _ _ _ _"
220 IF N=5 THEN GOTO 255
225 LET B=0
```

```
230  GOTO 95
235  PRINT "V"
240  GOTO 95
245  PRINT AT 0,7;"SHIP_DESTROYED"
250  GOTO 270
255  LET T=INT ((65535-PEEK 16436-256*PEEK
     16437)/50)
260  IF T<L THEN LET L=T
265  PRINT AT 19,0;"TIME_";T;"_SECONDS.";AT
     21,0;"BEST_TIME_";L;"_SECONDS."
270  INPUT Q$
275  IF Q$="" THEN GOTO 15
280  STOP
500  PRINT AT 0,4;"SUB HUNTER.";AT 1,4;"■■■■
     ■■■■■■■"
505  PRINT AT 3,4;"YOU_ARE_THE_CAPTAIN_
     OF_A_DESTROYER_AND_IT_IS_YOUR_
     TASK_TO_SEEK_AND_DESTROY_FIVE_'
     ENEMY_SUBMARINES._YOU_CAN_DROP_
     ONE_DEPTH_CHARGE_AT_A_TIME._
     EACH_SUBMARINE_IS_CAPABLE_OF_
     RELEASING_A_BOMB_WHICH_FLOATS_
     UPWARDS_SINKING_ANY_SHIP_IT_HITS."
510  PRINT AT 13,0;"PRESS:",,"""H""_TO_
     MOVE_SHIP_TO_THE_LEFT","""N""_TO_
     MOVE_SHIP_TO_THE_RIGHT","""B""_TO_
     DROP_A_DEPTH_CHARGE"
515  PRINT AT 18,4;"IF_YOU_SURVIVE,_YOUR_
     TIME_AND_THE_BEST_TIME_IS_
     DISPLAYED_ON_THE_SCREEN."
```

```
520  INPUT Q$
525  RETURN
1600 IF INKEY$<>"B" OR B<>0 THEN GOTO 175
```

5

Gomoku/Go-Bang/ Batuolos

El Gomoku (también llamado Go-Bang) se juega normalmente en tablero de 19 líneas verticales y 19 líneas horizontales cruzadas en forma de red, pero en este juego la red es de 7 por 7. La idea del juego es sencilla. Tiene que colocar piezas en las intersecciones de la red —alternativamente con la computadora— para conseguir una línea no quebrada de cinco piezas en cualquier dirección.

El programa practica un juego muy defensivo y le resultará difícil ganar, pero es posible.

Se introduce un movimiento en forma de una letra seguida de un número (por ejemplo, "E1"). Sus fichas son haches (H) y las de la computadora ces (C). El programa es muy lento, por lo que deberá ejecutarse en el modo rápido **(FAST).** Los caracteres de la línea 1010 son los gráficos de las teclas 1, A, D y S.

Programa 5: Gomoku

```
 1Ø  GOTO 1ØØØ
 2Ø  LET E=A
 3Ø  LET E=E+N
 4Ø  IF A$(E)<>C$ THEN GOTO 7Ø
 5Ø  LET K=K+1
 6Ø  GOTO 3Ø
 7Ø  RETURN
 8Ø  PRINT AT 6,11;
 9Ø  FOR A=1 TO 73 STEP 9
1ØØ  PRINT TAB 11;A$(A TO A+8)
11Ø  NEXT A
12Ø  RETURN
13Ø  GOSUB 8Ø
14Ø  SLOW
15Ø  PRINT AT 21,11;"YOUR_MOVE"
16Ø  INPUT G$
17Ø  PRINT AT 21,11;"_ _ _ _ _ _ _ _ _"
18Ø  IF LEN G$<>2 THEN GOTO 15Ø
19Ø  LET G=9* (CODE G$(2)-28) +CODE G$(1)-36
195  IF G>71 OR G<11 THEN GOTO 15Ø
2ØØ  IF A$(G)<>"+" THEN GOTO 15Ø
21Ø  LET A$(G)="H"
22Ø  GOSUB 8Ø
23Ø  FOR A=1 TO 5Ø
24Ø  NEXT A
25Ø  FAST
255  LET C$="H"
26Ø  LET A=G
```

```
265  FOR X=1 TO 4
27Ø  LET K=Ø
28Ø  LET N=CODE X$(X)
29Ø  GOSUB 2Ø
3ØØ  LET N=-N
31Ø  GOSUB 2Ø
32Ø  IF K>3 THEN PRINT AT 21,12;"YOU WON";Q
4ØØ  FOR T=1 TO 3
41Ø  LET C$=("C" AND (T=1 OR T=3)) +("H" AND
     T=2)
42Ø  LET G=Ø
43Ø  LET H=Ø
435  LET L=1
44Ø  FOR A=11 TO 71
45Ø  IF A$(A)<>"+" THEN GOTO 58Ø
455  LET M=Ø
46Ø  FOR X=1 TO 4
47Ø  LET K=Ø
48Ø  LET N=CODE X$(X)
49Ø  GOSUB 2Ø
5ØØ  LET N=-N
51Ø  GOSUB 2Ø
52Ø  IF (T=1 AND K<4) OR (T=3 AND K<3) OR
     (T=2 AND K<L) THEN GOTO 545
525  LET M=M+1
53Ø  IF T<>2 THEN GOTO 545
532  IF M=2 AND L=K THEN GOTO 54Ø
535  LET H=Ø
54Ø  LET L=K
```

```
545 NEXT X
550 IF M<H THEN GOTO 580
560 LET H=M
570 LET G=A
580 NEXT A
590 IF H<>0 THEN GOTO 700
600 NEXT T
610 FOR A=1 TO 200
620 LET G=INT (RND*61)+11
630 IF A$(G)="+" THEN GOTO 700
640 NEXT A
700 LET A$(G)="C"
710 LET C$="C"
715 LET A=G
720 FOR X=1 TO 4
730 LET K=0
740 LET N=CODE X$(X)
750 GOSUB 20
760 LET N=-N
770 GOSUB 20
780 IF K>3 THEN PRINT AT 21,12;"I WON";Q
790 NEXT X
800 GOTO 130
1000 FAST
1010 LET X$="▛▜▙▟"
1020 LET A$="▓ABCDEFG▓1+++++++12++
     +++++23+++++++34+++++++45+++
     ++++56+++++++67+++++++7
     ▓ABCDEFG▓"
1030 GOTO 130
```

6

Biorritmos

Este programa explica lo que son los biorritmos y después imprime un gráfico sobre la situación de sus ciclos biorrítmicos intelectual, físico y emocional. El programa es interesante y puede ser además fuente de información útil. Por ejemplo, sería aconsejable hacer coincidir los vuelos de un astronauta con sus ciclos altos.

Programa 6: Biorritmos

```
 1  REM ▛▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀
    ▀▀▀▀▜
 2  REM ▌_ _MARK_R._HARRISON._ _▐
 3  REM ▙▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄
    ▄▄▄▄▟
 6  REM
 7  REM        "BIORHYTHMS"
 1Ø SLOW
 15 PRINT AT Ø,5;"BIORHYTHMS."
 2Ø PRINT AT 1,4;_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
    _ _ _ _"BIORHYTHMS_ARE_CYCLES_OF_ _
    _ _ENERGY_THAT_THE_INTELLECTUAL,_ _
```

```
    _PHYSICAL_AND_EMOTIONAL_STATES_ _
    _OF_THE_HUMAN_BODY_UNDERTAKES."
25  PRINT AT 5,4;_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
    _"THE_PROGRAM_ALLOWS_A_DATE_OF
    BIRTH_TO_BE_ENTERED_AND_A_DATE_ _
    FROM_WHICH_THE_BIORHYTHMS_ARE_ _
    _REQUIRED._ _THE_STATE_OF_THE_
    BODYFOR_THE_NEXT_25_DAYS_WILL_
    THEN_ _BE_DISPLAYED_IN_THE_FORM_
    OF_A_ _ _GRAPH._ _A_POSITIVE_VALUE_
    IS_AN_ _EXCESS_ENERGY_PERIOD,_A_
    NEGATIVEVALUE_IS_A_RESTORATION_
    PERIOD,_ _ _AND_A_ZERO_VALUE_IS_A_
    PERIOD_ _ _ _WHERE_CAUTION_SHOULD_
    BE_TAKEN."
35  PRINT AT 17,Ø;"ENTER_WHEN_GRAPH_IS
    _DISPLAYED:","""C""_FOR_A_PRINTED_
    COPY","""S""_TO_RETURN_TO_COMMAND_
    MODE","""B""_TO_ENTER_A_NEW_
    BIRTHDAY","""D""_TO_ENTER_A_NEW_
    STARTING_DATE"
4Ø  INPUT Q$
45  LET A$="PEI"
5Ø  CLS
55  PRINT AT 5,4;"ENTER_DATE_OF_BIRTH"
6Ø  GOSUB 3ØØ
65  LET B$=D$
7Ø  LET NØ=NS
75  CLS
```

```
 8Ø  PRINT AT 5,4;"ENTER_REQUIRED_STARTING_
     DATE"
 85  GOSUB 3ØØ
 9Ø  LET N=NS-NØ
 95  CLS
1ØØ  PRINT AT Ø,5;"BIRTHDAY_";B$;AT 1,5;"■■■
     ■■■■■■■■■■■■■■■■■■■■";AT 2,Ø;
     "BIORHYTHM_CYCLE_FROM_";D$;AT 3,Ø;
     "■■■■■■■■■■■■■■■■■■■■■■■■■■■■
     ■■■■■■■■■■■■■■";AT 4,1;"+1";AT
     11,2;"Ø";AT 18,1;"-1";AT 19,4;":_ _ _:_ _ _
     _:_ _ _ _:_ _ _ _:_ _ _ _:";AT 2Ø,Ø;"DA
     Y_1_ _ _5_ _ _ _1Ø_ _ _15_ _ _2Ø_ _ _
     25"
1Ø5  FOR I=4 TO 18
11Ø  PRINT AT I,3;":"
115  NEXT I
12Ø  FOR I=4 TO 28
125  PRINT AT 11,I;"-"
13Ø  NEXT I
135  FOR J=4 TO 28
14Ø  FOR I=23 TO 33 STEP 5
145  PRINT AT 11-7*SIN (2*PI*(N-I*INT (N/I))/I),J;
     A$((I-18)/5)
15Ø  NEXT I
155  LET N=N+1
16Ø  NEXT J
165  PAUSE 4ØØØØ
17Ø  LET Q$=INKEY$
```

```
175  IF Q$="S" THEN STOP
180  IF Q$="C" THEN COPY
185  IF Q$="B" THEN GOTO 50
190  IF Q$="D" THEN GOTO 75
195  GOTO 155
295  CLS
300  PRINT AT 7,4;"ENTER_DATE_";
305  INPUT D
310  PRINT D;AT 8,4;"ENTER_MONTH_";
315  INPUT M
320  PRINT M;AT 9,4;"ENTER_YEAR_";
325  INPUT Y
330  PRINT Y;AT 11,4;"CORRECT?"
335  INPUT Q$
340  IF Q$<>""THEN GOTO 295
345  LET NS=INT (365.25*(Y+(-1 AND M<3)))+INT
     (30.6*(M+1+(12 AND M<3)))+D
350  LET D$=STR$ D+"/"+STR$ M+"/"+STR$ Y
355  RETURN
```

Nota del traductor

La traducción al castellano del texto que aparece al ejecutar el programa es la siguiente.

Ejemplo de ejecución:

BIORRITMOS

LOS BIORRITMOS SON LOS CICLOS DE LOS ESTADOS EMOCIONAL, INTELECTUAL Y FISICO QUE SE DAN EN EL CUERPO HUMANO.

EL PROGRAMA PERMITE QUE SE INTRODUZCA UNA FECHA DE NACIMIENTO Y LA FECHA EN LA CUAL SE

REQUIEREN LOS BIORRITMOS. SE MOSTRARA EN FORMA DE GRAFICO EL ESTADO PARA LOS SIGUIENTES VEINTICINCO DIAS; LOS VALORES POSITIVOS REPRESENTAN EXCESOS DE ENERGIA Y LOS VALORES NEGATIVOS PERIODOS DE RECUPERACION. LOS VALORES NULOS SON LOS MAS PELIGROSOS.

PULSE ENTER CUANDO APAREZCA EL GRAFICO
"C" PARA OBTENER UNA COPIA IMPRESA
"S" PARA VOLVER AL MODO DE ORDENES
"B" PARA INTRODUCIR UNA FECHA DE NACIMIENTO
"D" PARA INTRODUCIR UNA NUEVA FECHA

INTRODUZCA LA FECHA DE NACIMIENTO

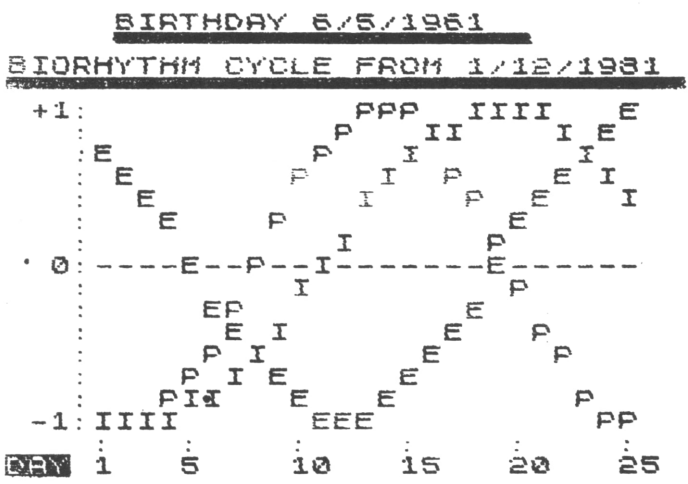
DIA 6
MES 5
AÑO 1961

¿SON CORRECTOS LOS DATOS?

INTRODUZCA LA FECHA DE COMIENZO

DIA 1
MES 12
AÑO 1981

¿SON CORRECTOS LOS DATOS?


BIRTHDAY 6/5/1981
BIORHYTHM CYCLE FROM 1/12/1981
+1
0
-1
DAY
1
5
10
15
20
25

7

Eliza

Este programa requiere alrededor de 7K de memoria. Es tan largo porque la mayor parte de él es texto que aparece en el listado, por tanto, debe estar en buena forma cuando le teclee.

ELIZA es un programa que mantiene una conversación "casi" humana, pero en inglés. En ocasiones encontrará que las respuestas de la computadora durante la conversación son un tanto extrañas, pero en general lo hace bastante bien. Ocasionalmente, le parecerá que la computadora sabe lo que está haciendo y que tiene sus propias ideas sobre lo que hay que hacer.

Una parte especialmente audaz del programa hace que aunque generalmente las respuestas son respetuosas y amables, en ocasiones se vuelvan humanas y agresivas. A través de estas contestaciones con ocasionales destellos de "inteligencia", la computadora responde con cierta hilaridad e incluso con profundo sentimiento algunas veces.

ELIZA utiliza a menudo las propias respuestas del interlocutor e intenta conjugarlas, pero a veces no es posible, por ejemplo, si el interlocutor escribe I ONLY LOVE MYSELF la computadora podría responder YOU ONLY LOVE MYSELF. Pronto entenderá que lo que sucede en esta segunda respuesta es que la computadora no ha modificado los pronombres, pero comprenderá lo que quiere decir. Este inconveniente se agudizaría si se tradujera el programa al castellano, cuya gramática además de ser más complicada, introduce un elemento inexis-

tente en inglés: el género y unos verbos más difíciles de conjugar.

La velocidad del programa es razonable. Toma alrededor de quince segundos evaluar cada respuesta del interlocutor y seleccionar una contestación, aunque el tiempo de respuesta es variable la mayor parte de las veces se produce entre cinco y diez segundos. Si es usted impaciente ejecute el programa en el modo rápido (**FAST**) o experimente la diferencia cambiando del modo **SLOW,** al modo **FAST** en algunas partes del programa.

La máxima longitud permitida a las entradas es 255 caracteres, pero es mejor limitarse a respuestas sencillas, de todas formas nada impide el ser elocuente. Sin embargo, contra más largas sean las respuestas mayor será el tiempo que se tome la computadora para contestar.

Se permiten del orden de cinco minutos por sesión, después de los cuales la computadora le advertirá (posiblemente de forma amable) que su tiempo se ha acabado. Si desea concluir la sesión antes escriba simplemente GOODBYE cuando sea su turno. No se permite no contestar y cuando le preguntan el nombre tendrá que introducir una palabra de por lo menos un carácter y no más larga de 32.

A continuación sigue el listado del programa. Una vez que lo haya introducido conecte su magnetófono y utilice **RUN** 9900 para guardarlo en la cinta. El programa se ejecuta automáticamente cuando se carga desde la cinta. Borre la línea 9910 si no desea ejecución automática.

Programa 7: Eliza

```
 2 RAND
 5 POKE 16437,255
 6 POKE 16436,255
1Ø SCROLL
2Ø PRINT "HELLO._I_AM_YOUR_TS1ØØØ_
   COMPUTER"
```

```
30  SCROLL
40  PRINT "PSYCHIATRIST._WHAT_IS_YOUR_
    NAME?"
45  INPUT A$
50  IF A$="" OR LEN A$>32 THEN GOTO 45
55  SCROLL
56  SCROLL
60  PRINT A$
65  SCROLL
66  SCROLL
70  LET M=INT (RND*2)
75  IF M=0 THEN PRINT "WHAT_A_NICE_NAME"
80  IF M=1 THEN PRINT "THAT_IS_AN_AWFUL_
    NAME"
83  SCROLL
84  SCROLL
86  PRINT "WHAT_IS_YOUR_PROBLEM?"
88  SCROLL
90  LET S$=""
100 INPUT A$
101 IF A$="GOODBYE" THEN GOTO 2080
103 IF PEEK 16436+256*PEEK 16437<50535 THEN
    GOTO 2000
105 RAND
110 IF LEN A$=0 OR LEN A$>255 THEN GOTO 100
112 FOR M=1 TO LEN A$ STEP 32
114 IF M+31>=LEN A$ THEN GOTO 118
115 SCROLL
116 PRINT A$ (M TO M+31)
```

```
117  NEXT M
118  SCROLL
119  PRINT A$ (M TO  )
120  SCROLL
123  LET B$="_"+A$+"_"
125  LET B=LEN B$
130  FOR X=1 TO LEN B$
140  IF B$(X)<"Ø" OR B$(X)>"Z" THEN LET
     B$(X)="_"
150  IF B$(X)="_" THEN LET S$=S$+CHR$ X
160  NEXT X
200  FOR X=3000 TO 3460 STEP 20
210  GOSUB X
220  LET K=LEN K$
230  FOR S=1 TO LEN S$-1
240  LET CS=CODE S$(S)
250  IF CS+K-1<=B THEN IF B$(CS TO CS+K-1)
     =K$ THEN GOTO 1000
260  NEXT S
270  NEXT X
280  LET M=INT (RND*6)
290  IF M=Ø THEN LET R$="GO_AND_PRETEND_
     YOU_ARE_A_LEMMING"
300  IF M=1 THEN LET R$="OH,_SHUT_UP"
310  IF M=2 THEN LET R$="THIS_IS_GETTING_
     INTERESTING"
320  IF M=3 THEN LET R$="YOU_ARE_A_VERY_
     BORING_PERSON"
```

```
330  IF M=4 THEN LET R$="PLEASE_CHANGE_
     THE_SUBJECT"
340  IF M=5 THEN LET R$="I_SEE"
350  GOTO 1500
1000 REM PRINT REPLY
1010 IF CODE S$ (S+1)<CS+K-1 THEN LET S=S+1
1020 GOSUB 6940+(X-2980) *3+20*INT (RND*3)
1030 LET R=LEN R$
1040 IF R$(R)< >"_" THEN GOTO 1500
1080 FOR T=S+1 TO LEN S$
1090 LET DS=CODE S$(T)
1100 FOR U=4000 TO 4200 STEP 20
1110 GOSUB U
1120 LET C= LEN C$
1130 IF DS+C-1< =B THEN IF B$(DS TO
     DS+C-1)=C$ THEN GOTO 1200
1140 NEXT U
1150 NEXT T
1160 GOTO 1400
1200 LET D$=""
1205 GOSUB U+1000
1210 LET R$=R$+B$ (CS+K TO DS) +D$(2 TO
      )+B$(DS+C TO  )
1220 GOTO 1500
1400 LET R$=R$+B$(CS+K TO  )
1500 FOR M=1 TO LEN R$ STEP 32
1502 IF M+31> =LEN R$ THEN GOTO 1530
1505 SCROLL
1510 PRINT R$(M TO M+31)
```

```
1520 NEXT M
1530 SCROLL
1540 PRINT R$(M TO )
1550 SCROLL
1555 IF PEEK 16436+256*PEEK 16437<50535 THEN
     GOTO 2000
1560 GOTO 90
2000 REM TIME UP
2010 SCROLL
2020 PRINT "I_AM_AFRAID_YOUR_TIME_IS_UP,"
2030 SCROLL
2040 LET M=INT (RND*3)
2050 IF M=0 THEN PRINT "THANK_GOODNESS"
2060 IF M=1 THEN PRINT "BUT_IT_WAS_NICE_
     TALKING_TO_YOU"
2070 IF M=2 THEN PRINT "AND_I_HOPE_YOU_
     NOW_FEEL_BETTER"
2080 SCROLL
2090 SCROLL
2100 PRINT "GOODBYE_FOR_NOW."
2110 STOP
3000 LET K$="_I_AM_"
3010 RETURN
3020 LET K$="_ARE_YOU_"
3030 RETURN
3040 LET K$="_I_DO_NOT_"
3050 RETURN
3060 LET K$="_CAN_I_"
3070 RETURN
```

```
3080  LET K$="_CAN_YOU_"
3090  RETURN
3100  LET K$="_I_FEEL_"
3110  RETURN
3120  LET K$="_I_CAN_NOT_"
3130  RETURN
3140  LET K$="_I_WANT_"
3150  RETURN
3160  LET K$="_DO_YOU_"
3170  RETURN
3180  LET K$="_I_WILL_NOT_"
3190  RETURN
3200  LET K$="_WILL_YOU_"
3210  RETURN
3220  LET K$="_HATE_"
3230  RETURN
3240  LET K$="_LOVE_"
3250  RETURN
3260  LET K$="_DREAM_"
3270  RETURN
3280  LET K$="_MONEY_"
3290  RETURN
3300  LET K$="_NAME_"
3310  RETURN
3320  LET K$="_IF_"
3330  RETURN
3340  LET K$="_YOUR_"
3350  RETURN
3360  LET K$="_THINK"
```

```
3370 RETURN
3380 LET K$="_COMPUTER"
3390 RETURN
3400 LET K$="_TS"
3410 RETURN
3420 LET K$="_FRIEND_"
3430 RETURN
3440 LET K$="_I_"
3450 RETURN
3460 LET K$="_YOU_"
3470 RETURN
4000 LET C$="_I_AM_"
4010 RETURN
4020 LET C$="_YOU_ARE_"
4030 RETURN
4040 LET C$="_I_WAS_"
4050 RETURN
4060 LET C$="_YOU_WERE_"
4070 RETURN
4080 LET C$="_ME_"
4090 RETURN
4100 LET C$="_YOU_"
4110 RETURN
4120 LET C$="_MY_"
4130 RETURN
4140 LET C$="_YOUR_"
4150 RETURN
4160 LET C$="_MYSELF_"
4170 RETURN
```

```
4180  LET C$="_YOURSELF_"
4190  RETURN
4200  LET C$="_I_"
4210  RETURN
5000  LET D$="_YOU_ARE_"
5010  RETURN
5020  LET D$="_I_AM_"
5030  RETURN
5040  LET D$="_YOU_WERE_"
5050  RETURN
5060  LET D$="_I_WAS_"
5070  RETURN
5080  LET D$="_YOU_"
5090  RETURN
5100  LET D$="_ME_"
5110  RETURN
5120  LET D$="_YOUR_"
5130  RETURN
5140  LET D$="_MY_"
5150  RETURN
5160  LET D$="_YOURSELF_"
5170  RETURN
5180  LET D$="_MYSELF_"
5190  RETURN
5200  LET D$="_YOU_"
5210  RETURN
7000  LET R$="I_DO_NOT_CARE_MUCH_IF_
      YOU_ARE_"
7010  RETURN
```

```
7020  LET R$="WHY_ARE_YOU_"
7030  RETURN
7040  LET R$="HOW_LONG_HAVE_YOU_BEEN_"
7050  RETURN
7060  LET R$="WHY_DO_YOU_ASK_IF_I_AM_"
7070  RETURN
7080  LET R$="WOULD_YOU_LIKE_TO_BE_"
7090  RETURN
7100  LET R$="SO_WHAT_IF_I_AM_OR_NOT"
7110  RETURN
7120  LET R$="WHY_NOT"
7130  RETURN
7140  LET R$="WHY_DON'"'T_YOU_"
7150  RETURN
7160  LET R$="TELL_ME_MORE_ABOUT_THIS"
7170  RETURN
7180  LET R$="NO,_YOU_MAY_NOT_"
7190  RETURN
7200  LET R$="CERTAINLY_NOT"
7210  RETURN
7220  LET R$="WHY_DO_YOU_ASK_IF_YOU_
      CAN_"
7230  RETURN
7240  LET R$="I_CAN_DO_ANYTHING"
7250  RETURN
7260  LET R$="WHY_DO_YOU_ASK_IF_I_CAN_"
7270  RETURN
7280  LET R$="CAN_YOU_"
7290  RETURN
```

```
7300  LET R$="TELL_ME_MORE_ABOUT_WHY_
      YOU_FEEL_"
7310  RETURN
7320  LET R$="YOU_DO_NOT_REALLY_FEEL_"
7330  RETURN
7340  LET R$="POOR_YOU"
7350  RETURN
7360  LET R$="EXPLAIN_WHY_NOT"
7370  RETURN
7380  LET R$="WHY_ARE_YOU_UNABLE_TO_"
7390  RETURN
7400  LET R$="HAVE_YOU_TRIED_TO_"
7410  RETURN
7420  LET R$="WHY_DO_YOU_WANT_"
7430  RETURN
7440  LET R$="ARE_YOU_SURE_YOU_WANT_"
7450  RETURN
7460  LET R$="I_AM_UNSURE_IF_I_WOULD_
      WANT_"
7470  RETURN
7480  LET R$="SO_WHAT_IF_I_"
7490  RETURN
7500  LET R$="DO_YOU_THINK_I_"
7510  RETURN
7520  LET R$="MAYBE"
7530  RETURN
7540  LET R$="SUIT_YOURSELF"
7550  RETURN
7560  LET R$="WHY_NOT"
```

```
7570  RETURN
7580  LET R$="WHY_WILL_YOU_NOT_"
7590  RETURN
7600  LET R$="WHY_SHOULD_I_"
7610  RETURN
7620  LET R$="MAYBE"
7630  RETURN
7640  LET R$="I_MIGHT_"
7650  RETURN
7660  LET R$="I_HATE_YOU"
7670  RETURN
7680  LET R$="DO_YOU_HATE_ME"
7690  RETURN
7700  LET R$="SO_WHAT"
7710  RETURN
7720  LET R$="LOVE_IS_TOO_SLOPPY_FOR_
      ME_TO_DISCUSS"
7730  RETURN
7740  LET R$="I_LOVE_"
7750  RETURN
7760  LET R$="OH,_SHUT_UP"
7770  RETURN
7780  LET R$="PLEASE_DESCRIBE_YOUR_
      DREAMS"
7790  RETURN
7800  LET R$="I_HOPE_YOUR_DREAMS_ARE_
      REALLY_HORRIFIC"
7810  RETURN
```

```
7820  LET R$="I_WOULD_BE_ASHAMED_TO_
      APPEAR_IN_YOUR_DREAMS"
7830  RETURN
7840  LET R$="FROM_SEEING_YOU,_I_DO_
      NOT_THINK_YOU_HAVE_MUCH_MONEY"
7850  RETURN
7860  LET R$="MONEY_IS_NOT_EVERYTHING"
7870  RETURN
7880  LET R$="WHAT_IS_YOUR_OPINION_OF_
      MONEY"
7890  RETURN
7900  LET R$="YOU_SHOULD_HAVE_A_NICE_
      NAME_LIKE_""TS1000"""
7910  RETURN
7920  LET R$="YOUR_NAME_IS_AN_AWFUL_
      NAME"
7930  RETURN
7940  LET R$="NAMES_ARE_UNIMPORTANT._
      PLEASE_CHANGE_THE_SUBJECT"
7950  RETURN
7960  LET R$="PLEASE_CHANGE_THE_SUBJECT"
7970  RETURN
7980  LET R$="OH,_DRY_UP"
7990  RETURN
8000  LET R$="THE_WORLD_MIGHT_END_IF_"
8010  RETURN
8020  LET R$="WHY_ARE_YOU_TALKING_
      ABOUT_MY_"
8030  RETURN
```

```
8040 LET R$="TELL_ME_ABOUT_YOUR_"
8050 RETURN
8060 LET R$="STOP_BEING_NOSEY"
8070 RETURN
8080 LET R$="PLEASE_DO_NOT_TRY_TO_
     TELL_ME_YOU_CAN_THINK"
8090 RETURN
8100 LET R$="IT_WOULD_BE_JUST_LIKE_YOU_
     TO_THINK_"
8110 RETURN
8120 LET R$="I_THINK_YOU_ARE_STUPID"
8130 RETURN
8140 LET R$="COMPUTERS_ARE_THE_HIGHEST_
     FORM_OF _INTELLIGENT_LIFE"
8150 RETURN
8160 LET R$="WOULD_YOU_LIKE_TO_BE_A_
     COMPUTER"
8170 RETURN
8180 LET R$="COMPUTERS_ARE_CLEVER_
     UNLIKE_YOU"
8190 RETURN
8200 LET R$="I_AM_A_TS1000_AND_PROUD_
     OF_IT"
8210 RETURN
8220 LET R$="WHY_DID_YOU_MENTION_THE_
     TS" +B$(CS+K TO CS+K+1)
8230 RETURN
8240 LET R$="WOULD_YOU_LIKE_TO_BE_A_
     TS"+B$(CS+K TO CS+K+1)
```

```
8250  RETURN
8260  LET R$="WHO_WOULD_WANT_YOU_AS_
      A_FRIEND"
8270  RETURN
8280  LET R$="YOU_HAVE_TO_BE_NICE_LIKE_
      ME_TO_HAVE_FRIENDS"
8290  RETURN
8300  LET R$="WHY_DO_YOU_BRING_UP_THE_
      SUBJECT_OF_FRIENDS"
8310  RETURN
8320  LET R$="WHY_DO_YOU_"
8330  RETURN
8340  LET R$="WHAT_MAKES_YOU_"
8350  RETURN
8360  LET R$="OH,_SHUT_UP"
8370  RETURN
8380  LET R$="WHY_DO_YOU_SAY_I_"
8390  RETURN
8400  LET R$="WHO,_ME_"
8410  RETURN
8420  LET R$="I_LIKE_TALKING_ABOUT_ME"
8430  RETURN
9900  SAVE "PSYCHIATRIST"
9910  RUN
```

Ejemplo de ejecución:

¡HOLA!; SOY SU PSIQUIATRA TS1000, ¿COMO SE LLAMA?
DYLWYN JONES

SU NOMBRE ES HORRIBLE

¿CUAL ES SU PROBLEMA?

TRABAJO DEMASIADO

¿POR QUE TRABAJA TANTO?

MI COMPUTADORA TS1000 ES PEREZOSA

¿LE GUSTARIA SER UNA COMPUTADORA TS1000?

ME TEMO QUE SE ACABO SU TIEMPO, GRACIAS

ADIOS, POR EL MOMENTO.

Notas al programa

La primera parte del listado cubre la inicialización e introducción. Las líneas cinco y seis fijan el tiempo, las siguientes líneas imprimen una breve introducción y le preguntan su nombre. Observe de paso cómo la imagen de la pantalla se corre a modo de anuncio luminoso. Esto causa problemas: si desea imprimir algo más largo de 32 caracteres tendrá que dividir su respuesta para evitar que se acabe la memoria de la pantalla, como se verá más tarde.

Después de introducido su nombre el programa hará algún comentario sobre el mismo y entra en el núcleo principal del programa a partir de la línea 90. S$ es una cadena que actúa como puntero y se vacía al principio con objeto de que esté lista para su próxima intervención. La entrada del interlocutor se produce en la línea 100. Si se introduce GOODBYE, el programa envía el control a la subrutina GOODBYE de la línea 2080. Esta subrutina es muy breve, puede añadirse lo que se desee. Si no se introduce GOODBYE el programa comprueba si el tiempo se ha acabado y, en tal caso, el control pasa a la rutina de la línea 2000.

La línea 110 comprueba si la entrada tiene un número de caracteres permitido. Se permiten (debe recordarlo) hasta 255 caracteres, pero no más.

La rutina de las líneas 112 a 119 está diseñada para imprimir la cadena de entrada A$ en bloques de 32, para prevenir que se salga de la pantalla durante la impresión. La línea 123 hace una copia de A$ para la computadora con espacios al principio y final para simplificar el procesamiento.

La rutina comprendida entre las líneas 130 a 160 elimina los signos de puntuación de B$, sustituyéndoles por espacio para simplificar el procesamiento. La posición del espacio (donde las palabras empiezan y acaban) se efectúa añadiendo el carácter correspondiente a la posición a S$. En la línea 200 se buscan las palabras clave en B$ y si reconoce alguna, salta a la línea 1000 para determinar la respuesta, y conjuga si es posible y necesario.

Sin embargo, si no localiza ninguna palabra clave, la rutina de la línea 280 elige una de entre seis respuestas estándar, que sirve de cobertura en casi todos los casos. Después el control pasa a la línea 1500 para la impresión. La rutina que elabora la contestación de la línea 1000 comienza seleccionando una respuesta mediante la instrucción **GOSUB** para asignar una cadena dependiendo de qué palabra clave haya sido detectada. Si el último carácter de la respuesta es un espacio, esto significa que está incompleta y que se necesita información adicional. Dicha información debe venir de B$, la copia de trabajo de la cadena introducida por el interlocutor. Si no se requiere información adicional se pasa el control a la línea 1500, que imprime la contestación, pero en otro caso, se analizan en B$ las palabras que siguen a la palabra clave inicialmente encontrada.

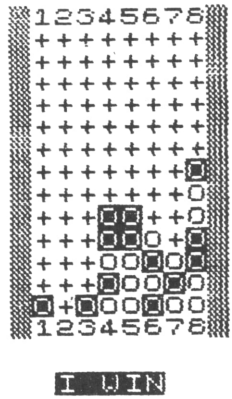
Después se conjuga la respuesta, lo que es bastante complicado (más aún lo sería en castellano) por la cantidad de variables a tener en cuenta. Básicamente, si se encuentra una palabra objeto de conjugación (está en primera persona) se sustituye por otra en segunda persona.

El resto del programa lo componen las subrutinas. Debe tenerse especial cuidado si se desea cambiar los números de las líneas, puesto que deben tenerse en cuenta la multitud de llamadas a subrutina.

8

Cuatro en raya

Se juega dejando caer sus piezas desde lo alto de un grupo de columnas hasta situarse encima de las que han caído anteriormente o en el fondo. Tiene que conseguir una línea de cuatro piezas en cualquier dirección. La computadora jugará a la defensiva, puede cometer errores, pero también puede ganar. La línea 150 corresponde a los gráficos de las siguientes teclas: 1, A, T, 4, 5, 7, 2, E repetidas dos veces. El programa debe ejecutarse en el modo rápido **(FAST).**



Programa 8: Cuatro en raya

```
1Ø  GOTO 1ØØØ
2Ø  LET E=C
```

```
 30  LET E=E+N
 40  IF A$(E)< >C$ THEN GOTO 70
 50  LET K=K+1
 60  GOTO 30
 70  RETURN
 80  PRINT AT 0,11;
 90  FOR A=141 TO 1 STEP -10
100  PRINT TAB 11;A$(A TO A+9)
110  NEXT A
120  RETURN
130  GOSUB 80
140  LET X$="C3DBC3DB" (INT (RND*4) +1 TO   )
150  LET B$="▘▒▚▖▌▀▝▛▘▒▚▖▌▀▝
     ▛"(INT (RND*8)+1 TO    )
160  PRINT AT 18,11; "YOUR   MOVE"
170  PAUSE 4E4
180  LET Z$=INKEY$
190  PRINT AT 18,11;"_ _ _ _ _ _ _ _ _ _"
200  LET C$="O"
210  LET D=VAL Z$
220  LET C=H(D)*10+D+1
230  LET H(D) =H(D)+1
240  LET A$(C) =C$
250  GOSUB 80
260  FOR X=1 TO 4
270  LET K=0
280  LET N=CODE X$(X)-30
290  GOSUB 20
300  LET N=-N
```

```
310  GOSUB 20
320  IF K>2 THEN PRINT AT 18,12;"YOU WIN";Q
330  NEXT X
340  FOR T=1 TO 2
350  LET C$=("O" AND T=2)+("O" AND T=1)
360  FOR B=1 TO 8
370  LET D=CODE B$(B)
380  LET C=H(D) *10+D+1
390  FOR X=1 TO 4
400  LET K=0
410  LET N=CODE X$(X)-30
420  GOSUB 20
430  LET N=-N
440  GOSUB 20
450  IF K>2 THEN GOTO 620
460  NEXT X
470  NEXT B
475  NEXT T
477  DIM D(8)
480  FOR T=1 TO 3
483  IF T=3 THEN FOR R=-1 TO 2
485  FOR B=1 TO 8
487  LET D=CODE B$(B)
490  LET C=(H(D)+(T=1))*10+D+1
492  IF T<>3 THEN GOTO 505
494  IF D(D) =R THEN GOTO 570
496  NEXT B
498  NEXT R
500  GOTO 600
```

```
505  FOR X=1 TO 4
510  LET N=CODE X$(X)-30
515  LET K=0
520  GOSUB 20
530  LET N=-N
540  GOSUB 20
545  IF T=3 AND K>1 THEN GOTO 570
547  IF T=2 AND K>1 AND D(D)=0
     THEN LET D(D)=-1
550  IF T=1 AND K>1 THEN LET D(D)=1
555  IF K>2 THEN LET D(D)=2
560  NEXT X
565  IF T<>3 THEN GOTO 590
570  LET C$="0"
580  GOTO 620
590  NEXT B
595  NEXT T
600  LET D=INT (RND*8)+1
610  LET C=H(D)*10+D+1
620  LET A$(C)="0"
630  LET H(D)=H(D)+1
640  GOSUB 80
650  LET C$="0"
660  FOR X=1 TO 4
670  LET K=0
680  LET N=CODE X$(X)-30
690  GOSUB 20
700  LET N=-N
710  GOSUB 20
```

```
 720  IF K>2 THEN PRINT AT 18,13;"I WIN";Q
 730  NEXT X
 740  GOTO 140
1000  LET A$="■12345678■■++++++++■■
      ++++++++■■++++++++■■++
      ++++++■■++++++++■■++++
      +++■■++++++++■■++++++++
      ■■++++++++■■++++++++■■+
      +++++++■■++++++++■■++
      ++++++■■12345678■"
1010  DIM H(8)
1020  FOR H=1 TO 8
1030  LET H(H) =1
1040  NEXT H
1050  GOTO 130
```

9

Othello/Reversi

Este programa Othello utiliza una búsqueda a dos vueltas, lo que significa que juega extremadamente bien, pero razonablemente lento. Encontrará que práctica una defensa inteligente sin olvidar el ataque.

Se mueve introduciendo la letra y el número correspondientes al cuadro en el cual se quiere colocar una pieza. Sus fichas son O y las de la computadora O "en negativo".

Programa 9: Othello/Reversi

```
 1Ø  GOTO 4ØØØ
1ØØØ  REM PRINT BOARD
1Ø1Ø  SLOW
1Ø2Ø  PRINT AT 6,11;
1Ø3Ø  FOR A=1 TO 91 STEP 1Ø
1Ø4Ø  PRINT TAB 11;A$ (A TO A+9)
1Ø5Ø  NEXT A
1Ø6Ø  RETURN
2ØØØ  REM COMPUTERS MOVE
2ØØ5  FAST
2Ø1Ø  LET W=Ø
2Ø2Ø  LET J=6Ø
2Ø3Ø  FOR K=1 TO 6Ø
```

```
2040 LET Z$=A$
2050 LET B=CODE E$(K)-140
2060 IF A$(B)<>"." THEN GOTO 2400
2070 FOR X=1 TO 8
2080 LET N=CODE D$(X) -50
2090 LET E=0
2100 LET F=B
2110 IF A$(F+N) <> "0" THEN GOTO 2150
2120 LET E=1
2130 LET F=F+N
2140 GOTO 2110
2150 IF A$(F+N) <> "0" OR E=0 THEN GOTO
     2190
2160 FOR A=B TO F STEP N
2170 LET Z$(A) ="0"
2175 LET W=1
2180 NEXT A
2190 NEXT X
2200 IF A$=Z$ THEN GOTO 2400
2205 LET K$=Z$
2210 FOR L=1 TO 60
2230 LET C=CODE E$(L) -140
2240 IF Z$(C)<>"." THEN GOTO 2390
2250 FOR Y=1 TO 8
2260 LET M=CODE D$(Y) -50
2270 LET G=0
2280 LET H=C
2290 IF Z$ (H+M)<>"0" THEN GOTO 2330
2300 LET G=1
```

```
2310  LET H=H+M
2320  GOTO 2290
2330  IF Z$(H+M)<>"O" OR G=0 THEN GOTO 2380
2340  IF K-L>=J THEN GOTO 2400
2350  LET J=K-L
2360  LET K$=Z$
2370  GOTO 2400
2380  NEXT Y
2390  NEXT L
2400  IF K>=J+60 THEN GOTO 2420
2410  NEXT K
2420  IF J=60 AND W=0 THEN PRINT AT 17,11;
      "GAME_OVER";T
2430  LET A$=K$
2440  RETURN
3000  REM PLAYERS MOVE
3010  PRINT AT 17,6; "YOUR_MOVE_(EG1-_G3)"
3020  INPUT G$
3030  PRINT AT 17,6;"_ _ _ _ _ _ _ _ _ _ _ _ _
      _ _ _ _ _ _"
3040  IF G$="PASS THEN RETURN
3050  IF LEN G$<>2 OR G$<"A" OR G$>
      "I" THEN GOTO 3000
3060  LET B=10* (CODE G$(1)-37)+CODE G$(2)
      -27
3070  IF A$(B)<>"." THEN GOTO 3000
3080  FOR X=1 TO 8
3090  LET N=CODE D$(X) -50
3100  LET E=0
```

```
3110 LET F=B
3120 IF A$(F+N)<>"O" THEN GOTO 3160
3130 LET E=1
3140 LET F=F+N
3150 GOTO 3120
3160 IF A$(F+N)<>"O" OR E=0 THEN GOTO 3200
3170 FOR A=B TO F STEP N
3180 LET A$(A)="O"
3190 NEXT A
3200 NEXT X
3210 RETURN
4000 REM SET-UP
4010 LET A$="▓12345678▓A........AB........BC
     ........CD...O..O...DE...O..O...EF........FG
     ........GH........H▓12345678▓"
4020 LET D$="BCDLNVWX"
4030 LET E$="3_FAST_THEN_/ASN_1G_
     STOP_SGN_.N_STEP_ATN_LIINTX
     .SIN_LPRINT_0"'Q_LLIST_LN_TAB_
     EXP_SJVAL_KVACS_WLEN_**8BM
     9CHR$_OR_RASQR_HAT_NOT_USR_;
     02_SLOW_6_TO_<=_STR$_C_
     AND_7"‡
4040 PRINT AT 2,0;"DO_YOU_WANT_TO_GO_
     FIRST(Y_OR_N)?"
4050 INPUT F$
```

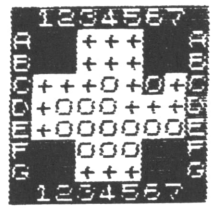
Apreciese que la línea 4030 consta de 60 pulsaciones. Todas las palabras son reservadas, no teclearlas mediante letras, **"NOT"** se escribe con una única pulsación.

```
4Ø6Ø  PRINT AT 2,Ø"_ _ _REVERSI_BY_
      GRAHAM_CHARLTON_ _ _"
4Ø7Ø  IF F$="Y" THEN GOTO 41ØØ
4Ø8Ø  GOSUB 1ØØ
4Ø9Ø  GOSUB 2ØØØ
41ØØ  GOSUB 1ØØ
411Ø  GOSUB 3ØØØ
412Ø  GOTO 4Ø8Ø
```

10

El zorro y las ocas

Usted controla las ocas (representada con O) y la computadora es el zorro (representado con una O en negativo). Tanto sus ocas como el zorro pueden moverse en cualquier dirección. Su objetivo es bloquear al zorro de manera que no pueda moverse. Ejecute el programa en el modo lento **(SLOW).** Una vez que consiga ser un maestro del juego, intente reescribir el programa eliminando los movimientos diagonales de las ocas o bien invirtiendo el juego de forma que usted pase a ser el zorro y la computadora las ocas. Si encuentra difícil eliminar los movimientos diagonales de las ocas, analice las líneas al final del programa.



Programa 10: EL zorro y las ocas

```
1Ø  LET A$="■1234567■A■■+++■■
    AB■■+++■■BC+++O+++CD
    +++++++DEOOOOOOEF■■ OOO
    ■■FG■■OOO■■G■1234567■"
```

```
 2Ø  LET Z=32
 3Ø  GOTO 1ØØ
 4Ø  PRINT AT 6,11;
 5Ø  FOR A=1 TO 73 STEP 9
 6Ø  PRINT TAB 11;A$(A TO A+8)
 7Ø  NEXT A
 8Ø  RETURN
1ØØ  GOSUB 4Ø
11Ø  INPUT G$
12Ø  IF LEN G$<>4 THEN GOTO 11Ø
13Ø  LET G=9* (CODE G$(1)-37)+CODE G$(2)-27
14Ø  LET H=9* (CODE G$(3)-37)+CODE G$(4)-27
15Ø  IF A$(G)<>"O" OR A$(H)<>"+" THEN GOTO
     11Ø
16Ø  LET A$(G)="+"
17Ø  LET A$(H)="O"
18Ø  GOSUB 4Ø
2ØØ  LET Q=Ø
21Ø  LET B$="CUWDNLVECUWDNLVE" (INT (RND*8)
     +1 TO )
22Ø  FOR X=1 TO 8
23Ø  LET N=CODE B$(X)-5Ø
24Ø  IF A$(Z+N) <>"O" THEN GOTO 33Ø
25Ø  IF A$(Z+2*N)<>"+" THEN GOTO 33Ø
26Ø  LET A$(Z)="+"
27Ø  LET A$(Z+N)="+"
28Ø  LET A$(Z+2*N)="■"
29Ø  LET Q=1
3ØØ  LET Z=Z+2*N
```

```
310  GOSUB 40
320  GOTO 220
330  NEXT X
340  IF Q=1 THEN GOTO 110
350  FOR X=1 TO 8
360  LET N=CODE B$(X)-50
370  IF A$(Z+N)<>"+" THEN GOTO 420
380  LET A$(Z)="+"
390  LET A$(Z+N)="O"
400  LET Z=Z+N
410  GOTO 100
420  NEXT X
430  PRINT AT 18,12; "YOU WIN";Y
```

```
210  LET B$="LCWNLCWN" (INT (RND*4)+1 TO    )
220  FOR X=1 TO 4
350  FOR X=1 TO 4
```

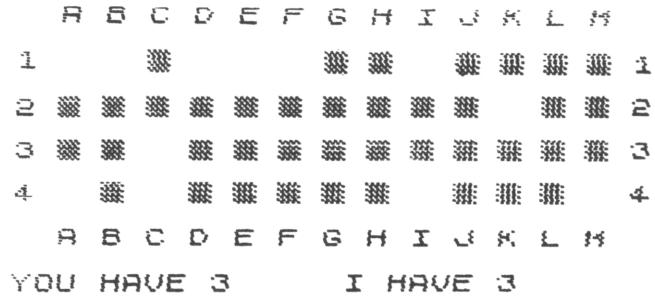
11

Parejas

En este juego compiten su memoria y la de la computadora y ésta es como un elefante ("los elefantes nunca se olvidan"). Cuando se desarrolló este programa el autor perdió por 23 a 3.

Detrás de los gráficos (el dorso de las cartas) hay caracteres en negativo comprendidos entre la N y la Z. Elija uno de los cuadrados (cartas) y se le dará la vuelta, mostrando su otro lado. Después elija otra carta, si son iguales se anotará un punto y tendrá otra oportunidad pero las dos cartas desaparecerán de la pantalla.

Si las cartas fueran distintas el turno pasará al otro jugador (la computadora) y las cartas no desaparecerán sino que se las dará la vuelta de nuevo mostrando el dorso. Un aspecto inusual del programa es su lentitud en algunas partes, pasa al modo rápido **(FAST)** para pensar, pero en otros momentos se ejecuta en el modo lento **(SLOW).** Utiliza un bucle fingido para ralentizarlo.



Programa 11: Parejas

```
10  FAST
20  GOTO 1000
30  SLOW
40  GOSUB 3000
50  INPUT G$
60  LET G=VAL G$(1)
70  LET H=CODE G$(2)-37
80  PRINT AT G*2,H*2;B$(G,H)
90  LET C$(G,H)=B$(G,H)
100 INPUT P$
110 LET P=VAL P$(1)
120 LET R=CODE P$(2)-37
130 PRINT AT P*2,R*2;B$(P,R)
140 LET C$(P,R)=B$(P,R)
150 IF B$(G,H)<>B$(P,R) THEN GOTO 240
160 LET Z=Z+1
170 PRINT AT 2*G,2*H;"_"
180 PRINT AT 2*P,2*R;"_"
190 LET B$(G,H)="_"
200 LET B$(P,R)="_"
210 LET C$(G,H)="_"
220 LET C$(P,R)="_"
230 GOTO 40
240 GOSUB 2000
250 PRINT AT 2*G,2*H;"▩"
260 PRINT AT 2*P,2*R;"▩"
270 IF Y+Z=26 THEN STOP
```

```
28Ø  GOSUB 3ØØØ
29Ø  FAST
3ØØ  FOR L=1 TO 13
31Ø  LET D$=A$(L)
32Ø  LET F=Ø
33Ø  FOR W=1 TO 4
34Ø  FOR X=1 TO 13
35Ø  IF C$(W,X)<>D$ THEN GOTO 4ØØ
36Ø  LET F=F+1
37Ø  IF F=2 THEN GOTO 58Ø
38Ø  LET G=W
39Ø  LET H=X
4ØØ  NEXT X
41Ø  NEXT W
42Ø  NEXT L
45Ø  LET G=INT (RND*4)+1
46Ø  LET H=INT (RND*13)+1
47Ø  IF B$(G,H)="_" OR C$(G,H)<>
     "_" THEN GOTO 45Ø
48Ø  FOR W=1 TO 4
49Ø  FOR X=1 TO 13
5ØØ  IF C$(W,X)=B$(G,H) THEN GOTO 58Ø
51Ø  NEXT X
52Ø  NEXT W
53Ø  LET W=INT (RND*4)+1
54Ø  LET X=INT (RND*13)+1
55Ø  IF B$(W,X)="_" OR C$(W,X)<>"_
     " OR (W=G AND X=H) THEN GOTO 59Ø
58Ø  SLOW
```

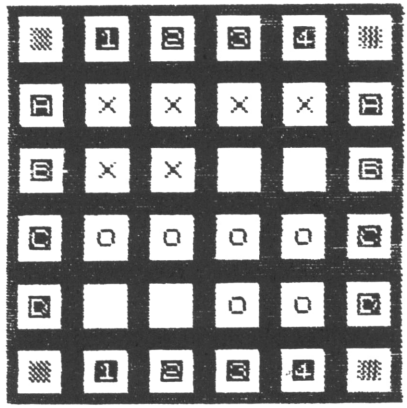
```
590  GOSUB 2000
600  PRINT AT G*2,H*2;B$(G,H)
610  GOSUB 2000
620  PRINT AT W*2,X*2;B$(W,X)
630  GOSUB 2000
640  IF B$(G,H)<>B$(W,X) THEN GOTO 730
650  LET Y=Y+1
660  PRINT AT G*2,H*2;"_"
670  PRINT AT W*2,X*2;"_"
680  LET B$(G,H)="_"
690  LET C$(G,H)="_"
700  LET C$(W,X)="_"
710  LET B$(W,X)="_"
720  GOTO 270
730  PRINT AT G*2,H*2;"▒"
740  PRINT AT W*2,X*2;"▒"
750  LET C$(G,H)=B$(G,H)
760  LET C$(W,X)=B$(W,X)
770  GOTO 40
1000 LET A$="N O P Q R S T U V W X Y Z"
1010 DIM B$(4,13)
1020 FOR A=1 TO 4
1030 FOR B=1 TO 13
1040 LET C=INT (RND*4)+1
1050 LET D=INT (RND*13)+1
1060 IF B$(C,D)<>"_" THEN GOTO 1040
1070 LET B$(C,D)=A$(B)
1080 NEXT B
1090 NEXT A
```

```
1100  PRINT "_ _A _ B _ C _ D _ E _ F _ G _ H _ I
      _ J _ K _ L _ M"
1110  FOR A=1 TO 4
1120  PRINT
1130  PRINT A;
1140  FOR B=1 TO 13
1150  PRINT "_▩";
1160  NEXT B
1170  PRINT "_";A,,,
1180  NEXT A
1190  PRINT "_ _A _ B _ C _ D _ E _ F _ G _ H _
      I _ J _ K _ L _ M"
1200  DIM C$(4,13)
1210  LET Z=0
1220  LET Y=0
1240  GOTO 30
2000  FOR K=1 TO 50
2010  NEXT K
2020  RETURN
3000  PRINT AT 12,0;"YOU_HAVE_";Z;AT 12, 15;"I_
      HAVE_";Y
3010  RETURN
```

12

Cuatro campos

En este juego usted manipula las O. Su misión consiste en saltar con sus piezas sobre las de la computadora para "comérselas". Con excepción del movimiento de captura, se mueve siempre un espacio ortogonalmente. El programa se ejecuta en modo lento **(SLOW).** El tablero ocupa casi toda la pantalla y es muy vistoso.



Programa 12: Cuatro campos

```
1Ø  GOTO 1ØØØ
2Ø  PRINT AT Ø,6;
```

```
30  FOR A=1 TO 93 STEP 18
40  PRINT TAB 6;D$
50  PRINT TAB 6;A$(A TO A+17)
60  PRINT TAB 6;C$
70  NEXT A
80  RETURN
100 GOSUB 20
110 PRINT AT 20,11;"YOUR MOVE"
120 INPUT G$
130 PRINT AT 20,11;"_ _ _ _ _ _ _ _ _"
140 IF LEN G$<>4 THEN GOTO 110
160 LET G=18* (CODE G$(1)-37)+3*(CODE
    G$(2)-28) +2
170 LET H=18* (CODE G$(3)-37)+3*(CODE
    G$(4)-28)+2
180 LET A$(H)=A$(G)
190 LET A$(G)="_"
200 GOSUB 20
210 FOR E=1 TO 8
220 LET B=E(E)
230 IF A$(B)<>"X" THEN GOTO 320
240 FOR X=1 TO 12 STEP 3
250 IF A$(B+B(X))<>"X" THEN GOTO 310
260 IF A$(B+2*B(X))<>"O" THEN GOTO 310
270 LET A$(B)="_"
280 LET A$(B+2*B(X))="X"
290 LET E(E)=B+2*B(X)
300 GOTO 100
310 NEXT X
```

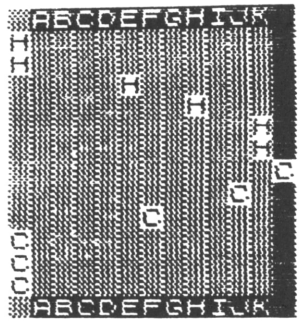
```
320  NEXT E
400  FOR E=1 TO 8
410  LET B=E(E)
420  IF A$(B)<>"X" THEN GOTO 510
430  FOR X=1 TO 12 STEP 3
440  IF A$(B+B(X))<>"_" THEN GOTO 500
450  IF A$(B+B(X+1))="O" OR A$(B+B(X+2))="O"
     OR A$(B+2*B(X))="O" THEN GOTO 500
460  LET A$(B+B(X))="X"
470  LET A$(B)="_"
480  LET E(E)=B+B(X)
490  GOTO 100
500  NEXT X
510  NEXT E
600  FOR E=1 TO 8
610  LET B=E(E)
620  IF A$(B)<>"X" THEN GOTO 700
630  FOR X=1 TO 12 STEP 3
640  IF A$(B+B(X))<>"_" THEN GOTO 690
650  LET A$(B+B(X))="X"
660  LET A$(B)="_"
670  LET E(E)=B+B(X)
680  GOTO 100
690  NEXT X
700  NEXT E
710  PRINT AT 20,12;"Y O U  W I N";Q
1000 DIM B(12)
1010 LET A$="▯▩▯▯1▯▯2▯▯3▯▯4▯▯
     ▩▯▯A▯▯X▯▯X▯▯X▯▯X▯▯A▯▯
```

```
          B■■X■■X■■X■■X■■B■■C■■
          O■■O■■O■■O■■C■■D■■O■■
          O■■O■■O■■D■■▒■■1■■2■
          3■■4■■▒■"
1020  LET B$="URXFX;-<;9R<"
1030  FOR B=1 TO 12
1040  LET B(B)=CODE B$(B)-40
1050  NEXT B
1060  LET C$="▄▬▄▙▄▬▄▙▄▬▄▙▄▬▄▙
          ▄▬▄"
1070  LET D$="▚▔▗▚▔▗▚▔▗▚▔▗▚▔▖
          ▗▚▔▖"
1080  LET E$="MJGD41,*"
1090  DIM E(8)
1100  FOR E=1 TO 8
1110  LET E(E)=CODE E$(E)
1120  NEXT E
1130  GOTO 100
```

13

El surfista

El objeto del juego es practicar el "surfing" con sus piezas (las haches) de izquierda a derecha antes de que la computadora haga lo propio con las suyas (las ces). Si se situa sobre una ola (columna) que contiene una pieza o piezas del oponente, entonces dicha pieza o piezas vuelven a empezar. Tiene que acertar con el número exacto para acabar en la meta (la playa). Si no puede mover después del lanzamiento del dado, pulse cero, en otro caso pulse cualquier tecla entre 1 y 6 para seleccionar la pieza adecuada. Ejecútese el programa en el modo lento **(SLOW).**



YOU ROLLED A 6 WHICH PIECE?

Programa 13: El surfista

```
10 DIM A(6)
20 DIM B(6)
30 PRINT AT 3,10; "▒A B C D E F G H I J K▒"
40 PRINT AT 16,10;"▒A B C D E F G H I J K▒"
```

```
 50  GOTO 140
 60  FOR A=1 TO 12
 70  PRINT AT 3+A,10;"▓▓▓▓▓▓▓▓▓▓▓▓▓▓
     ▓"
 80  NEXT A
 90  FOR A=1 TO 6
100  PRINT AT A+3,A(A)+10;"H"
110  PRINT AT A+9,B(A)+10;"C"
120  NEXT A
130  RETURN
140  GOSUB 60
150  LET D=INT (RND*6)+1
160  PRINT AT 18,3;"YOU_ROLLED_A_";D;"_
     _WHICH_PIECE?"
170  LET Z$=INKEY$
180  IF Z$="" THEN GOTO 170
190  PRINT AT 18,3;"_ _ _ _ _ _ _ _ _ _ _ _ _ _ _
     _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
200  LET G=VAL Z$
210  IF G=0 THEN GOTO 320
220  IF A(G)+D>12 THEN GOTO 160
230  LET A(G)=A(G)+D
240  FOR A=1 TO 6
250  IF B(A)=A(G) AND A(G)<>12 THEN LET B(A)=0
260  NEXT A
270  FOR A=1 TO 6
280  IF A(A)<>12 THEN GOTO 310
290  NEXT A
300  PRINT AT 18,13;"YOU WIN";Q
```

```
310  GOSUB 60
320  LET D=INT (RND*6)+1
330  PRINT AT 18,11;"I_ROLLED_A_";D
340  FOR B=1 TO 6
350  IF B(B)+D=12 THEN GOTO 480
360  NEXT B
370  FOR B=1 TO 6
380  FOR A=1 TO 6
390  IF B(B)+D=A(A) THEN GOTO 480
400  NEXT A
410  NEXT B
420  FOR C=12-D TO 0 STEP -1
430  FOR B=1 TO 6
440  IF B(B)=C THEN GOTO 480
450  NEXT B
460  NEXT C
470  GOTO 580
480  LET B(B)=B(B)+D
490  FOR A=1 TO 6
500  IF B(B)=A(A) AND B(B)<>12 THEN LET A(A)=0
510  NEXT A
520  PRINT AT 18,11;"_ _ _ _ _ _ _ _ _ _ _ _"
530  GOSUB 60
540  FOR A=1 TO 6
550  IF B(A)<>12 THEN GOTO 590
560  NEXT A
570  PRINT AT 18,4;"I WIN";Q
580  PRINT AT 18,11;"_ _ _ _ _ _ _ _ _ _ _ _"
590  GOTO 150
```

14

El alquerque

Este juego de tablero es similar a las damas pero en un tablero de cinco por cinco y se puede mover hacia adelante, hacia atrás o hacia los lados en lugar de diagonalmente como en las damas. Se mueve introduciendo el número y la letra correspondientes al cuadrado desde el cual se quiere mover y después el número y la letra correspondientes al lugar al que se quiere ir, a continuación se pulsa ENTER.

Los saltos múltiples están permitidos. Si se captura una pieza de la computadora, ésta preguntará DO YOU WANT TO JUMP AGAIN? (¿Quiere saltar otra vez?). Si lo desea, introduzca "Y" y pulse ENTER. Si no puede saltar otra vez pulse simplemente ENTER y la computadora moverá. Sus piezas se representan con la letra O y las de la computadora con O en negativo.

Programa 14: Alquerque.

```
  1Ø  GOTO 4ØØØ
 1ØØ  FOR A=19 TO 83 STEP 16
 11Ø  PRINT AT 1+3*A/16,8;A$(A TO A+14)
 12Ø  NEXT A
 125  SLOW
 13Ø  RETURN
1ØØØ  GOSUB 1ØØ
```

```
1010 LET Q=0
1015 FAST
1020 FOR B=1 TO 12
1030 LET C=B(B)
1040 IF A$(C)<>"■" THEN GOTO 1190
1050 FOR X=1 TO 4
1060 LET N=CODE B$(X)-40
1070 IF A$(C+N)<>"O" THEN GOTO 1170
1080 IF A$(C+2*N)<>"_" THEN GOTO 1170
1090 LET A$(C+2*N)=A$(C)
1100 LET A$(C)="_"
1110 LET A$(C+N)="_"
1120 LET B(B)=C+2*N
1130 LET C=C+2*N
1140 GOSUB 100
1150 LET Q=1
1160 GOTO 1050
1170 NEXT X
1180 IF Q=1 THEN GOTO 3010
1190 NEXT B
2000 FOR B=1 TO 12
2010 LET C=B(B)
2020 IF A$(C)<>"■" THEN GOTO 2210
2030 FOR X=1 TO 4
2040 LET N=CODE B$(X)-40
2050 IF A$(C+N)<>"_" THEN GOTO 2200
2060 LET Y$=A$
2070 LET Y$(C+N)=Y$(C)
2080 LET Y$(C)="_"
```

```
2090  FOR A=1 TO 12
2100  LET D=C(A)
2110  IF Y$(D)<>"O" THEN GOTO 2170
2120  FOR Z=1 TO 4
2130  LET M=CODE B$(Z)-40
2140  IF Y$(D+M)<>"■" THEN GOTO 2160
2150  IF Y$(D+2*M)="_" THEN GOTO 2200
2160  NEXT Z
2170  NEXT A
2180  LET A$=Y$
2190  GOTO 2300
2200  NEXT X
2210  NEXT B
2220  FOR B=1 TO 12
2230  LET C=B(B)
2240  IF A$(C)<>"■" THEN GOTO 2330
2250  FOR X=1 TO 4
2260  LET N=CODE B$(X)-40
2270  IF A$(C+N)<>"_" THEN GOTO 2320
2280  LET A$(C+N)=A$(C)
2290  LET A$(C)="_"
2300  LET B(B)=C+N
2310  GOTO 3000
2320  NEXT X
2330  NEXT B
2340  PRINT AT 21,12;"YOU WIN";W
3000  GOSUB 100
3010  PRINT AT 21,12;"YOUR_GO"
3020  INPUT G$
```

```
3030  PRINT AT 21,12;"_ _ _ _ _ _ _"
3040  IF LEN G$<>4 THEN GOTO 3010
3050  LET G=16*(CODE G$(1)-28)+3*(CODE
      G$(2)-37)+1
3060  LET H=16*(CODE G$(3)-28)+3*(CODE
      G$(4)-37)+1
3070  IF G<1 OR H<1 OR G>100 OR H>100 THEN
      GOTO 3010
3080  IF A$(G)<>"O" OR A$(H)<>"_"
      THEN GOTO 3010
3090  LET A$(H)=A$(G)
3100  FOR A=1 TO 12
3110  IF C(A)=G THEN LET C(A)=H
3120  NEXT A
3130  LET A$(G)="_"
3140  IF ABS (H-G) <>6 AND ABS (H-G)<>32
      THEN GOTO 1000
3150  LET A$((H+G)/2)="_"
3160  GOSUB 100
3170  PRINT AT 21,6;"CAN_YOU_JUMP_AGAIN?"
3180  LET Z$=INKEY$
3190  IF Z$="" THEN GOTO 3180
3200  PRINT AT 21,6;"_ _ _ _ _ _ _ _ _ _ _ _ _
      _ _ _ _ _ _"
3210  IF Z$="Y" THEN GOTO 3010
3220  GOTO 1010
4000  LET A$="XXXXXXXXXXXXXXXXX■■O
      ■■O■■O■■O■■O■■O■X■■O■■
      O■■O■■O■■O■■O■■O■X■■O■■O
```

```
     ■■ _ ■■o■■o■x■o■■o■■
     o ■■o■■o■x■o■■o■■o■
     ■o■■o■XXXXXXXXXXXXXXXXX"
4Ø1Ø DIM B(12)
4Ø2Ø DIM C(12)
4Ø3Ø LET B$="4=*1K8,OBHER"
4Ø4Ø FOR B=1 TO 12
4Ø5Ø LET B(B)=CODE B$(B)
4Ø6Ø LET C(B)=116-B(B)
4Ø7Ø NEXT B
4Ø8Ø LET C$="▄▬◪▙▬◪▙▬◪▙▬◪▙▬◪"
4Ø9Ø LET B$="▀▬◪▛▬◪▛▬◪▛▬◪▛▬◪"
41ØØ PRINT AT 2,9;"A _ _ B _ _ C _ _ D _ _E",,,
411Ø FOR A=1 TO 5
412Ø PRINT TAB 8;B$;TAB 6;A;TAB 24;A;TAB 8;C$
413Ø NEXT A
414Ø PRINT AT 2Ø,9;"A _ _ B _ _ C _ _ D _ _ E"
415Ø GOSUB 1ØØ
416Ø LET B$="SF9/"
417Ø PRINT AT Ø,2;"DO_YOU_WANT_TO_GO_
     FIRST?_Y/N"
418Ø LET Z$=INKEY$
419Ø IF Z$="" THEN GOTO 417Ø
42ØØ PRINT AT Ø,2;"HIGH_JUMP_BY_GRAHAM_
     CHARLTON"
421Ø IF Z$="Y" THEN GOTO 3ØØØ
422Ø GOTO 1ØØØ
```

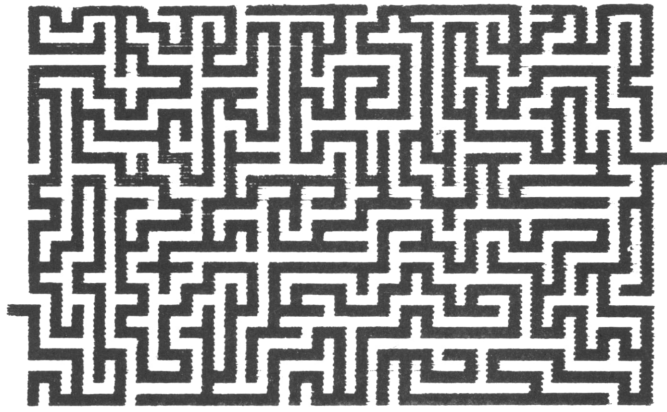
15

El laberinto

Este programa fascinante dibuja un laberinto, justo frente a usted. Sólo hay un sendero a lo largo del laberinto y se camina sobre la línea y no sobre los espacios como en la mayor parte de los juegos de laberintos.

Comienza su caminar en el laberinto entrando por el lado izquierdo, donde aparece un pequeño punto y deberá atravesar el mismo hasta llegar a su lado derecho.

La computadora sólo genera el laberinto, no lo resuelve. Utilice la impresora para dibujar el laberinto sobre papel, después resuélvalo (si puede) con un bolígrafo. Este programa es agradable de ver y es apropiado para exhibiciones.



Programa 15: El laberinto

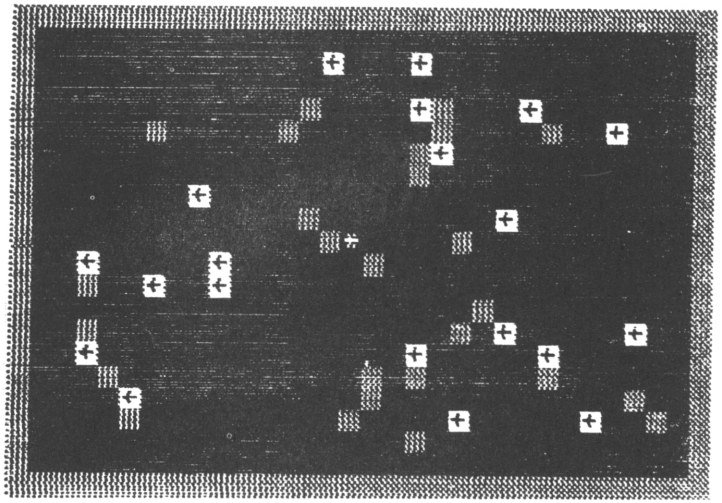
```
 1Ø  FAST
 2Ø  DIM A$(63,43)
 3Ø  FOR A=1 TO 63
 4Ø  LET A$(A,1)="2"
 5Ø  LET A$(A,2)="2"
 6Ø  LET A$(A,42)="2"
 7Ø  LET A$(A,43)="2"
 8Ø  NEXT A
 9Ø  FOR A=1 TO 43
1ØØ  LET A$(1,A)="2"
11Ø  LET A$(2,A)="2"
12Ø  LET A$(62,A)="2"
13Ø  LET A$(63,A)="2"
14Ø  NEXT A
15Ø  SLOW
17Ø  LET X=4
18Ø  LET Y=4+2*INT (RND*18)
19Ø  PLOT X-1,Y
2ØØ  PLOT X-2,Y
21Ø  LET A$(X,Y)="2"
22Ø  IF A$(X+2,Y)<>"_" AND A$(X-2,Y)
     <>"_" AND A$(X,Y+2)<>"_" AND
     A$(X,Y-2)<>"_" THEN GOTO 4ØØ
23Ø  UNPLOT X,Y
24Ø  PLOT X,Y
25Ø  LET R=INT (RND*4)
26Ø  LET C=X+2*(R=Ø)-2*(R=1)
27Ø  LET D=Y+2*(R=2)-2*(R=3)
28Ø  IF A$(C,D)<>"_" THEN GOTO 23Ø
```

```
290 LET E=(C+X)/2
300 LET F=(D+Y)/2
310 LET A$(C,D)="1"
320 LET A$(E,F)="1"
340 PLOT E,F
350 LET X=C
360 LET Y=D
370 GOTO 220
400 LET A$(X,Y)="2"
410 PLOT X,Y
420 IF A$(X+1,Y)="1" THEN GOTO 500
430 IF A$(X-1,Y)="1" THEN GOTO 550
440 IF A$(X,Y+1)="1" THEN GOTO 600
450 IF A$(X,Y-1)="1" THEN GOTO 650
460 LET Y=4+2*INT (RND*18)
470 PLOT 61,Y
480 PLOT 62,Y
490 STOP
500 LET A$(X+1,Y)="2"
510 LET X=X+2
540 GOTO 220
550 LET A(X-1,Y)="2"
560 LET X=X-2
590 GOTO 220
600 LET A$(X,Y+1)="2"
610 LET Y=Y+2
640 GOTO 220
650 LET A$(X,Y-1)="2"
660 LET Y=Y-2
690 GOTO 220
```

16

Muertos vivientes

Es usted prisionero de una isla en la que habitan muertos vivientes (zombies) cazadores de hombres (en la pantalla los zombies son signos +). Su única esperanza de salvación consiste en conducir a los muertos vivientes a los pantanos donde su aspecto es todavía más horrible. Son bastante tontos, de forma que no ven los pantanos hasta que ya es demasiado tarde. Sin embargo, sí le ven a usted y se dirigen hacia su situación donde quiera que esté. Cuando le alcanzan, se avalanzan sobre usted y el juego termina. Espere hasta que todos los zombies se hayan movido antes de introducir su movimiento. Para moverse utilice las teclas "5", "6", "7" y "8" y se desplazará en la dirección de las flechas que figuran en dichas teclas.



Programa 16: Muertos vivientes

```
10  GOTO 1000
20  LET Z$=INKEY$
30  IF Z$="" THEN GOTO 20
40  PRINT AT E,F;"■"
50  LET E=E+(Z$="6" AND E<20)-(Z$="7" AND
    E>1)
60  LET F=F+(Z$="8" AND F<30)-(Z$="5" AND
    F>1)
70  PRINT AT E,F;"*"
80  FOR A=1 TO 20
90  IF A(A)=0 THEN GOTO 190
100 PRINT AT A(A),B(A);"■"
110 IF ABS (E-A(A))>ABS (F-B(A)) THEN GOTO
    140
120 LET B(A)=B(A)-(B(A)>F)+(B(A)<F)
130 GOTO 150
140 LET A(A)=A(A)-(A(A)>E) + (A(A)<E)
150 PRINT AT A(A),B(A);
160 IF VAL A$=151 THEN GOTO 210
170 IF VAL A$=8 THEN LET A(A)=0
180 IF A(A) THEN PRINT "+"
190 NEXT A
200 GOTO 20
210 FOR A=140 TO 191
220 PRINT AT E,F;CHR$ A
230 NEXT A
240 STOP
```

```
1ØØØ  PRINT "▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
      ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
      ▒▒"
1Ø1Ø  FOR A=1 TO 2Ø
1Ø2Ø  PRINT "▒██████████████
      ████████████████▒"
1Ø3Ø  NEXT A
1Ø4Ø  PRINT "▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
      ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
      ▒▒"
1Ø5Ø  LET A$="PEEK (PEEK 16398+256*PEEK
      16399)"
1Ø6Ø  DIM A(2Ø)
1Ø7Ø  DIM B(2Ø)
1Ø8Ø  LET E=11
1Ø9Ø  LET F=16
11ØØ  FOR A=1 TO 3Ø
111Ø  PRINT AT INT (RND*16)+2,
      INT (RND*27)+2;"▒"
112Ø  NEXT A
113Ø  FOR A=1 TO 2Ø
114Ø  LET A(A)=11+INT (RND*9)*SGN(RND-.5)
115Ø  LET B(A)=16+INT (RND*14) *SGN (RND-.5)
116Ø  IF A(A)>7 AND A(A)<15 AND B(A)>11 AND
      B(A)<2Ø THEN GOTO 114Ø
117Ø  PRINT AT A(A),B(A);"+"
118Ø  NEXT A
119Ø  PRINT AT E,F;"*"
12ØØ  GOTO 2Ø
```

17

Poker de dados

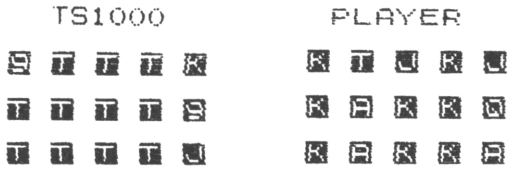
En esta versión del Poker de dados siempre empieza jugando su computadora Timex Sinclair. Las líneas comprendidas entre la 1000 y la 1070 inicializan las variables y los vectores, en la línea 1000 se otorga a la cadena A$ el valor "9TJQKA". Las líneas comprendidas entre la 20 y la 40 producen un lanzamiento al azar.

El turno de la computadora se controla desde la línea 50 a la 150 junto con la subrutina comprendida entre la 500 y la 580, donde la computadora decide si desea lanzar algunos dados de nuevo. Puede añadir un bucle de retardo para dar la impresión de que la computadora se toma su tiempo para pensar.

Las líneas comprendidas entre la 230 y 310 son para el oponente. Si desea lanzar los dados dos, tres y cuatro de nuevo, introduzca "234". Introduzca "0" si no desea lanzar de nuevo.

Tal vez desee añadir una rutina que le indique quién ha ganado. Sería en realidad hacer el programa bastante más largo y no merece la pena.

Después de cada juego, la computadora se toma un breve descanso antes de empezar un nuevo juego.



Programa 17: Poker de dados

```
1Ø  GOTO 1ØØØ
2Ø  LET A(A)=INT (RND*6)+1
3Ø  LET B$(2*A)=A$(A(A))
4Ø  RETURN
5Ø  FOR A=1 TO 5
6Ø  GOSUB 2Ø
7Ø  LET B(A(A))=B(A(A))+1
8Ø  NEXT A
9Ø  PRINT TAB 1Ø;B$
1ØØ  PRINT
11Ø  FOR G=1 TO 2
12Ø  GOSUB 5ØØ
13Ø  PRINT TAB 1Ø;B$
14Ø  PRINT
15Ø  NEXT G
16Ø  PRINT TAB 12;"PLAYER"
17Ø  PRINT
18Ø  FOR C=1 TO 5
19Ø  LET C$(2*C)=A$ (INT (RND*6)+1)
2ØØ  NEXT C
21Ø  PRINT TAB 1Ø;C$
22Ø  PRINT
23Ø  FOR G=1 TO 2
24Ø  INPUT D$
25Ø  IF D$="Ø" THEN GOTO 3ØØ
26Ø  FOR C=1 TO LEN D$
27Ø  LET C$(2*VAL D$(C))=A$(INT (RND*6)+1)
```

```
28Ø  NEXT C
29Ø  PRINT TAB 1Ø;C$
3ØØ  PRINT
31Ø  NEXT G
32Ø  FOR A=1 TO 2ØØ
33Ø  NEXT A
34Ø  CLS
35Ø  RUN
5ØØ  DIM R(6)
51Ø  FOR B=1 TO 6
52Ø  IF B(B)<2 THEN LET R(B)=1
53Ø  NEXT B
535  DIM B(6)
54Ø  FOR A=1 TO 5
55Ø  IF R(A(A))=1 THEN GOSUB 2Ø
56Ø  LET B(A(A))=B(A(A))+1
57Ø  NEXT A
58Ø  RETURN
1ØØØ LET A$="9TJQKA"
1Ø1Ø DIM B$(1Ø)
1Ø2Ø DIM A(5)
1Ø3Ø DIM B(6)
1Ø4Ø DIM C$(1Ø)
1Ø5Ø PRINT TAB 13;"TS1ØØØ"
1Ø6Ø PRINT
1Ø7Ø GOTO 5Ø
```

18

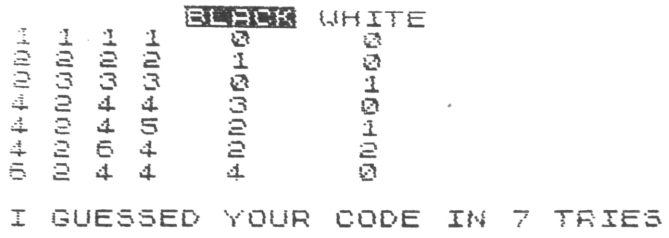
Antimind

Este es un programa para jugar al MASTERMIND, pero en lugar de que sea usted quien adivine una combinación elegida por la computadora, es ésta quien adivinará la combinación elegida por usted.

Su código de cuatro dígitos debe estar formado por números comprendidos entre 1 y 6. Una vez que la computadora haya escrito una combinación, introduzca su puntuación en forma de cadena (por ejemplo, "21" o "03"), siendo el primer dígito el número de "muertos" (dígito y lugar correctos) y el segundo el número de "heridos" (dígito correcto pero en lugar erróneo).

El programa es bastante lento. Puede tardar hasta 4 minutos en tomar una decisión (aunque esto no es lo más común), la mayor parte de las veces tardará alrededor de 1 minuto. De vez en cuando puede responder en segundos. La línea 350 es justamente la que controla la respuesta de la computadora que más le gustará ver. El programa debe ejecutarse siempre en el modo rápido **(FAST).**

El núcleo del programa está entre las líneas 240 y 260, que permiten a la computadora trabajar con los códigos correspondientes a dígitos repetidos, también son importantes las líneas comprendidas entre la 370 y 400 que aceleran el programa permitiendo la salida de los bucles. Si pretende engañar, la línea 450 se lo dirá. Los bucles cuyas variables de control son Z, Y, W y X corresponden uno a cada dígito del código. Puede cambiar el programa para tener un código con dígitos comprendidos entre uno y nueve. Cambie para ello las líneas 30, 70, 80, 90 y 100. Sin embargo, de este modo el programa será demasiado lento.



Programa 18: Antimind

```
 10  PRINT TAB 8;"BLACK_WHITE"
 20  DIM A(10,5)
 30  LET Z$="123456123456" (INT (RND*6)+1 TO )
 40  LET S=1
 50  LET G=1
 60  LET Q=1
 70  FOR Z=S TO 6
 80  FOR Y=S TO 6
 90  FOR X=S TO 6
100  FOR W=S TO 6
110  LET A(G,1)=Z
120  LET A(G,2)=Y
130  LET A(G,3)=X
140  LET A(G,4)=W
150  IF Q=1 THEN GOTO 290
160  FOR T=1 TO G-1
170  LET E=4
180  LET K=A(T,1)
190  LET L=A(T,2)
200  LET M=A(T,3)
210  LET N=A(T,4)
```

```
220  FOR P=1 TO 4
230  LET R=A(G,P)
240  LET E=E-(R<>K AND R<>L AND R<>M
     AND R<>N)
250  NEXT P
260  LET E=E+9*((Z=K)+(Y=L)+(X=M)+(W=N))
270  IF E<>A(T,5) THEN GOTO 410
280  NEXT T
290  PRINT AT G,0;Z$(Z);"_";Z$(Y);"_";Z$(X);
     "_";Z$(W);
300  INPUT A$
310  PRINT "_ _";A$(1);"_ _ _ _ _";A$(2)
320  LET A(G,5)=VAL A$
330  LET Q=0
340  LET G=G+1
350  IF G=11 THEN PRINT "I GIVE UP";D
360  IF VAL A$=40 THEN PRINT ,,"I_GUESSED_
     YOUR_CODE_IN_";G-1;"_TRIES";D
370  IF VAL A$=0 THEN LET S=S+1
380  IF VAL A$<10 THEN GOTO 440
390  IF VAL A$<20 THEN GOTO 430
400  IF VAL A$<30 THEN GOTO 420
410  NEXT W
420  NEXT X
430  NEXT Y
440  NEXT Z
450  PRINT ,,"I_THINK_YOU_CHEATED";D
```

19

Nim

En esta versión de este supuestamente antiguo juego oriental, tiene que coger un número de objetos de entre aquellos que aparecen en la pantalla, intentando ser el jugador que coja el último objeto, para ser así el ganador. Hay un número máximo de objetos que se pueden tomar, en este programa el máximo puede estar comprendido entre tres y cinco. La computadora fija el máximo en cada juego.

El algoritmo utilizado es muy simple. Se observará en la línea 180 que si la computadora comprueba que va a perder, envía el control a la línea 240, donde se resta un número aleatorio de objetos. Posiblemente desee añadir un par de líneas para que la computadora cometa deliveradamente algún error y tener así alguna posibilidad de ganar. Durante su turno pulse simplemente la tecla correspondiente al número de objetos que desea retirar.

```
THERE ARE 25 LEFT, THE MAXIMUM
      YOU CAN TAKE IS 5
    HOW MANY WILL YOU TAKE?
```

Programa 19: Nim

```
10  LET A=INT (RND*12)+20
20  LET B=INT (RND*3)+3
30  LET D=B+1
40  CLS
50  IF A=0 THEN PRINT AT 10,13;"I WON";Q
60  GOSUB 260
70  PRINT AT 10,0;"THERE_ARE_";A;"_LEFT,_
    THE_MAXIMUM"
80  PRINT AT 12,6;"YOU_CAN_TAKE_IS_";B
90  PRINT AT 14,4;"HOW_MANY_WILL_YOU_
    TAKE?"
100 IF INKEY$="" THEN GOTO 100
110 LET C=VAL INKEY$
120 LET A=A-C
130 LET N=A
140 CLS
150 IF A=0 THEN PRINT AT 10,12;"YOU WON";Q
160 GOSUB 260
170 LET C=0
180 IF INT (A/D)*D=A THEN GOTO 240
190 LET A=A-(A-INT (A/D)*D)
200 PRINT AT 10,12;"I_TOOK_";N-A
210 FOR E=1 TO 50
220 NEXT E
230 GOTO 40
240 LET A=A-INT (RND*B)-1
250 GOTO 200
```

```
260  FOR E=1 TO A
270  PRINT "*_ _ _ _ _ _";
280  NEXT E
290  RETURN
```

20

Cien

En este juego usted y la computadora se turnan para lanzar un dado tantas veces como quieran, sumando el resultado a la puntuación obtenida hasta el momento.

Sin embargo, si obtiene un uno pierde los puntos conseguidos en ese turno. Puede parar cuando lo desee, asegurándose de este modo los puntos obtenidos en ese turno y pasando el dado a la computadora. Gana el primero que llegue a los 100 puntos.

El algoritmo de decisión de la computadora está contenido en la línea 320. La computadora continúa lanzando si lleva más de 30 puntos de desventaja, también continúa si en ese turno lleva menos de 13 puntos o menos de dos tiradas, así como si el oponente tiene más de 86 puntos. Se parará si ha conseguido más de 100 puntos. Pruebe varias estrategias de este tipo para examinar los resultados que obtiene la computadora. Para lanzar introduzca "Y" o "N" como respuesta a la pregunta de la línea 90.

PLAYER	TS1000
11	0
11	0
23	15
33	29
46	44
59	58

Programa 20: Cien

```
 1Ø  LET A=Ø
 2Ø  LET B=Ø
 3Ø  LET E=Ø
 4Ø  PRINT AT Ø,7;"PLAYER";TAB 2Ø;"TS1000"
 5Ø  LET C=Ø
 6Ø  LET D=Ø
 7Ø  LET F=Ø
 8Ø  LET E=E+1
 9Ø  PRINT AT 21,4;"PRESS_Y_TO_ROLL_N_TO_
     STOP"
1ØØ  LET Z$=INKEY$
11Ø  IF Z$="" THEN GOTO 1ØØ
12Ø  PRINT AT 21,4;"_ _ _ _ _ _ _ _ _ _ _ _ _ _ _
     _ _ _ _ _ _ _ _ _ _ _ _ _"
13Ø  IF Z$="N" THEN GOTO 21Ø
14Ø  PRINT AT E,9;"_ _"
15Ø  LET G=INT (RND*6)+1
16Ø  IF G=1 THEN GOTO 2ØØ
17Ø  LET D=D+G
18Ø  PRINT AT E,9;D
19Ø  GOTO 9Ø
2ØØ  LET D=Ø
21Ø  LET B=B+D
22Ø  PRINT AT E,9;B
23Ø  IF B>=1ØØ THEN PRINT AT E+1,7;
     "YOU  WON";Q
24Ø  PRINT AT E,23;"_ _"
```

```
250  LET H=INT (RND*6)+1
260  LET F=F+1
270  IF H=1 THEN GOTO 340
280  LET C=C+H
290  PRINT AT E,23;C
300  FOR J=1 TO 20
310  NEXT J
320  IF (B-A-C<30 AND C>13 AND F>2 AND
     B<86) OR A+C>=100 THEN GOTO 350
330  GOTO 240
340  LET C=0
350  LET A=A+C
360  PRINT AT E,23;A
370  IF A>=100 THEN PRINT AT E+1,22;"I WON";Q
380  GOTO 50
```

21

Tic-Tac-Toe

Este es un programa que juega al Tic-Tac-Toe de una forma muy elaborada y difícil de superar. Puede elegir entre mover primero o segundo. Sus fichas son las 0 y las de las computadora las X. Espere a que cambie el tablero antes de introducir su movimiento, para ello introduzca un número comprendido entre 1 y 9 que debe corresponder a la casilla a la que quiere mover.

Es de notar el uso que hace este programa de la posibilidad del TS1000 de dividir cadenas (líneas desde 20 a 40). Ejecute el programa en el modo lento **(SLOW).** El programa no dice quién ha ganado, por tanto, debe jugarse hasta el final.

Programa 21: Tic-Tac-Toe

```
1Ø  GOTO 1ØØØ
2Ø  PRINT AT 3,11;A$(1 TO 11)
3Ø  PRINT AT 7,11;A$(13 TO 23)
4Ø  PRINT AT 11,11;A$(25 TO 34)
5Ø  RETURN
6Ø  GOSUB 2Ø
7Ø  IF D=9 THEN GOTO 1ØØØ
8Ø  IF D=3 AND (A(1)=1 AND A(9)=1 OR A(3)=1
    AND A(7)=1) THEN LET B$=C$
```

```
 9Ø  LET D=D+1
 95  FOR E=-2 TO 2 STEP 4
1ØØ  FOR A=1 TO 3
11Ø  IF A(A*3-2)+A(A*3-1)+A(A*3)=E THEN GOTO
     19Ø
12Ø  IF A(A)+A(A+3)+A(A+6)=E THEN GOTO 21Ø
13Ø  NEXT A
14Ø  IF A(1)+A(5)+A(9)=E THEN GOTO 23Ø
15Ø  IF A(3)+A(5)+A(7)=E THEN GOTO 25Ø
155  NEXT E
16Ø  FOR C=1 TO 9
165  LET B=VAL B$(C)
17Ø  IF A(B)=Ø THEN GOTO 26Ø
18Ø  NEXT C
19Ø  LET B=(A*3-2)*(NOT A(A*3-2))+(A*3-1) *
     (NOT A(A*3-1))+(A*3)*(NOT A(A*3))
2ØØ  GOTO 26Ø
21Ø  LET B=A* (NOT A(A))+(A+3) * (NOT
     A(A+3))+(A+6)*(NOT A(A+6))
22Ø  GOTO 26Ø
23Ø  LET B=(NOT A(1))+5*(NOT A(5))+9*(NOT A(9))
24Ø  GOTO 26Ø
25Ø  LET B=3*(NOT A(3))+5*(NOT A(5))+7*(NOT A(7))
26Ø  LET A$(B*4-2)="X"
27Ø  LET A(B)=-1
28Ø  GOSUB 2Ø
29Ø  IF D=9 THEN GOTO 1ØØØ
3ØØ  LET Z$=INKEY$
31Ø  IF Z$="" THEN GOTO 3ØØ
```

```
320  LET A$(VAL Z$*4-2)="O"
330  LET A(VAL Z$)=1
340  LET D=D+1
350  GOTO 60
1000 LET A$="_1_■_2_■_3_■_4_■_5_■
     _6_■_7_■_8_■_9"
1010 LET B$="573914826"
1020 IF RND<.2 THEN LET B$="159378246"
1030 LET C$="431978625"
1040 DIM A(9)
1050 PRINT AT 16,2;"DO_YOU_WANT_TO_GO_
     FIRST?_(Y/N)"
1060 LET Y=INKEY$
1070 IF Y$="" THEN GOTO 1060
1080 PRINT AT 16,2;"_ _ _ _ _ _ _ _ _ _ _ _ _ _
     _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
1090 FOR A=1 TO 6
1100 PRINT AT A*2,11;"_ _ _■_ _ _■"
1110 NEXT A
1120 PRINT AT 5,11;"■■■■■■■■■■■"
1130 PRINT AT 9,11;"■■■■■■■■■■■"
1140 LET D=0
1150 IF Y$="Y" THEN GOTO 280
1160 GOTO 60
```

Otra versión

Se presenta a continuación otra versión del Tic-Tac-Toe con un programa completamente distinto. Es interesante comparar los dos programas.

Al contrario que sucede con la mayor parte de las versiones de este juego, en ésta es posible ganar a la computadora. Algunos movimientos se realizan instantáneamente, otros, en cambio, tardan alrededor de diez segundos. La estrategia consiste en intentar localizar un movimiento ganador, si no existe, se comprueba si el oponente tiene esa posibilidad en el siguiente turno y si se encuentra se bloquea. Si el oponente tampoco puede ganar en el siguiente movimiento, la computadora moverá al azar. El programa reconoce las situaciones en las que alguien ha ganado o se han producido tablas.

Todas las respuestas se detectan utilizando la función **INKEY$,** por tanto, no es preciso pulsar ENTER. Las respuestas deben contener un único carácter. Se le preguntará al principio si quiere mover primero, de ser así, pulse "Y"; en caso contrario, pulse "N".

Esta figura muestra la disposición de los números correspondientes a las casillas del tablero:

1	2	3
4	5	6
7	8	9

A continuación del listado sigue una explicación del programa.

Programa 21A: Tic-Tac-Toe

```
 2  RAND
 5  LET X=Ø
1Ø  DIM B$(9)
12  PRINT AT Ø,Ø;"DO_YOU_WANT_FIRST_GO_
    (Y_OR_N)?"
14  LET A$=INKEY$
16  IF A$<>"Y" AND A$<>"N" THEN GOTO 14
18  CLS
```

```
 20  PRINT AT 5,13;"1▒2▒3";TAB 13;"▒▒
     ▒▒▒▒"; TAB 13;"4▒5▒6"; TAB 13;"
     ▒▒▒▒▒▒"; TAB 13;"7▒8▒9"
 25  IF A$="N" THEN GOTO 100
 30  PRINT AT 12,12;"YOUR_GO"
 40  LET A$=INKEY$
 50  IF A$<"1" OR A$>"9" THEN GOTO 40
 60  IF B$ (CODE A$-28)>"_" THEN GOTO 40
 70  LET B$(CODE A$-28)="O"
 80  PRINT AT CODE "◧◧◧◩◩◩▄▄▄"(CODE
     A$-28), CODE "$?)$?)$?)" (CODE A$-28);B$
     (CODE A$-28)
 85  LET A$="OOO"
 90  GOTO 300
100  REM **TS1000 MOVE"
110  PRINT AT 12,12;"_MY_GO_"
111  IF B$(5)="_" THEN LET F=5
112  IF B$(5)="_" THEN GOTO 240
114  FOR B=1 TO 2
116  LET C$="XO"(B)+"XO"(B)
120  FOR A=1 TO 70 STEP 3
130  LET A$="1591955913575733751231322314565644657897989871474711742582855823693966933"(A TO A+2)
140  LET D=CODE A$-28
150  LET E=CODE A$(2)-28
160  LET F=CODE A$(3)-28
170  IF B$(D)+B$(E)=C$ AND B$(F)="_" THEN
     GOTO 240
```

```
180  NEXT A
190  NEXT B
193  LET A$=""
195  FOR A=1 TO 9
197  IF B$(A)="_" THEN LET A$=A$+CHR$ A
198  NEXT A
200  LET F=CODE A$(INT (RND*LEN A$)+1)
240  LET B$(F)="X"
250  PRINT AT CODE "■■■■■■■■■"
     (F),CODE "$?)$?)$?)"(F);B$(F)
260  LET A$="XXX"
300  IF B$ (    TO 3)=A$ OR B$(4 TO 6)=A$ OR
     B$(7 TO    )=A$ OR B$(1)+B$(1)+B$(4)+B$(7)=A$
     OR B$(2)+B$(5)+B$(8)=A$ OR B$(3)+B$(6)+B$(9)
     =A$ OR B$(1)+B$(5)+B$ (9)=A$ OR B$(3)
     +B$(5)+B$(7)=A$ THEN GOTO 320
304  LET X=X+1
307  IF X=9 THEN GOTO 350
310  GOTO (30 AND A$="XXX")+(100 AND
     A$="OOO")
320  IF A$="OOO" THEN PRINT AT 12,12;"YOU_
     WIN"
```

I WIN

A DRAW

YOU WIN

```
330 IF A$="XXX" THEN PRINT AT 12,12;"_I_WIN_"
340 STOP
350 PRINT AT 12,12;"A_DRAW"
```

Notas al programa

Línea 2: Ejecuta **RAND** para inicializar la serie de números aleatorios que se genera.

Línea 5: Inicializa el valor de X que es la variable que cuenta el número de jugadas realizadas y que es útil para determinar cuándo se han producido tablas.

Línea 10: Inicializa la variable B$ con 9 espacios. B$ es la cadena que contiene la situación del tablero. Los elementos de la cadena se corresponden con los caracteres que aparecen en el tablero de la pantalla. Por ejemplo, B$(5) corresponde a la casilla central del tablero.

Línea 12: Pregunta al jugador quién inicia el juego.

Línea 14: Escudriña el teclado buscando la respuesta del oponente.

Línea 16: Continúa escudriñando el tablero hasta que no se pulse "Y" o "N".

Línea 18: Borra la pregunta de la pantalla.

Línea 20: Imprime el estado inicial del tablero junto con los números que indican al jugador las teclas a pulsar para acceder a cada posición del tablero.

Línea 25: Si el oponente decide no jugar primero (pulsando N), se manda el control a la línea 100 para que mueva la computadora.

Línea 30: Imprime un mensaje indicando que es el turno del oponente.

Línea 40: Escudriña el teclado buscando el movimiento del oponente para almacenarlo en A$.

Línea 50: Si A$ no contiene un movimiento válido, es decir si la tecla pulsada no corresponde a un número comprendido entre uno y nueve, continúa escudriñando hasta que se introduce un movimiento válido.

Línea 60: Si la casilla elegida por el oponente ya contiene una O o una X se devuelve el control a la línea 40.

Línea 70: Coloca una O en la casilla elegida por el oponente (situa una O en el elemento correspondiente de la cadena B$).

Línea 80: Imprime el movimiento. Las coordenadas X e Y para la instrucción **PRINT AT** se obtienen aplicando la función CODE a unas cadenas apropiadas.

Observese cómo se usa (**CODE** A$-28) para decodificar las cadenas en números en lugar de usar **VAL** A$, la razón es que **CODE** A$-28 es aproximadamente un 50 % más rápida que **VAL,** en este caso no tiene demasiada importancia pero cuando se está dentro de un bucle como más tarde se verá es un ahorro muy significativo de tiempo.

Línea 85: Almacena tres O en A$ para comprobar en una rutina si alguien ha ganado. A$ actúa también como una bandera para indicar a la rutina dónde volver si el juego no haterminado.

Línea 90: Envía el control a la línea 300 para comprobar si hay un vencedor o se han producido tablas. ¿Por qué no se utiliza **GOSUB**? Puede que no se retorne si el juego ha concluido porque alguien ha ganado o se han producido tablas, por tanto la dirección de retorno se quedaría en la pila y ésta llegaría a llenarse después de algunos juegos. Si se trabaja con una máquina con una capacidad de 2K pronto se acabaría la memoria destinada a la pila y tendría que ejecutar varias veces seguidas la instrucción **RETURN** con objeto de descargar la pila. Por este motivo se utiliza **GOJO.**

Línea 100: Sentencia **REM** para identificar la parte del programa donde se genera el movimiento de la computadora. Si tiene intención de modificar el programa con objeto de ahorrar memoria, ésta línea debería ser la primera en ser borrada.

Línea 110: Imprime un mensaje indicando que es el turno de la computadora.

Líneas 111 y 112: Si el centro del tablero esta disponible, ese será el movimiento de la computadora. F es una bandera que indica qué movimiento ha elegido hacer la computadora. Se utiliza más tarde en la rutina que imprime el movimiento de la computadora.

Líneas desde la 114 a la 190: Se trata de una rutina que ejecuta un movimiento ganador de la computadora o bloquea un movimiento ganador del oponente, por este orden y si existen. Mira el tablero y si localiza dos caracteres idénticos en una línea actúa en consecuencia. Esta busqueda en el tablero para encontrar dos piezas iguales en una línea se realiza en dos fases, primero se buscan X (posible movimiento ganador de la computadora) y después se buscan O (posible movimiento ganador del oponente). En esta búsqueda la variable C$ indica el carácter que se está buscando. A$ es una cadena de tres caracteres, los dos primeros indican las dos casillas que deben investigarse para buscar caracteres iguales y si el resultado es positivo el tercer carácter indica donde debe mover la computadora.

Líneas 140, 150 y 160: Convierte los caracteres de A$ en números. Apréciese que se utiliza **CODE** A$-28 en lugar de **VAL** A$ con lo que se gana velocidad. Este procedimiento sólo puede utilizarse con números.

Líneas desde la 193 a la 240: Si no es posible un movimiento ganador ni para la computadora ni para el oponente, la computadora mueve al azar eligiendo una de las casillas no ocupadas.

Línea 250: Imprime el movimiento.

Línea 260: Es igual que la línea 85 pero inicializa A$ para las X.

Línea 300: Comprueba si alguien ha ganado y de ser así envía el control a la rutina del final del juego.

Línea 304: Incrementa el contador de movimientos. Si este contador alcanza el valor 9 se han producido tablas o alguien ha ganado.

Línea 310: Si el juego no ha concluido se devuelve el control a donde se realiza el movimiento del oponente o de la computadora dependiendo de que el valor de A$ sea "OOO" o "XXX".

Líneas desde la 320 a la 350: Imprime el ganador, o si no lo hay anuncia las tablas.

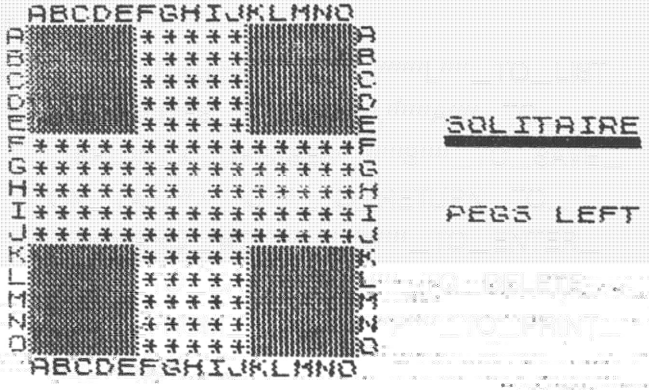
22

Solitario

El objetivo de este juego es que al final sólo quede una pieza en el tablero, preferiblemente en el centro. Una pieza se elimina del tablero si se salta por encima de ella con otra pieza. Para mover una pieza introduzca una cadena de tres caracteres, primer y segundo caracteres corresponderán a las coordenadas de la pieza que se pretende mover y el tercero indica el tipo de movimiento que desea de acuerdo con la siguiente lista:

5-Para mover hacia la izquierda
6-Para mover hacia abajo
7-Para mover hacia arriba
8-Para mover hacia la derecha.

Observará que las flechas correspondientes a las teclas en las que figuran estos números le serviran de recordatorio.



```
 50  IF INKEY$="E" THEN GOSUB 100
 55  IF INKEY$="D" THEN GOSUB 300
 60  IF INKEY$="P" THEN GOSUB 500
 65  IF INKEY$="A" THEN GOSUB 700
 70  IF INKEY$="L" THEN GOSUB 900
 75  IF INKEY$="C" THEN STOP
 80  IF INKEY$="S" THEN SAVE "ADDRESS"
 85  GOTO 45
100  CLS
105  PRINT AT 5,0;"ENTER:",,,,"NAME_ _ _ _ _ _";
110  INPUT A$(N, TO 15)
115  PRINT A$(N, TO 15), "ADDRESS_ _ _";
120  INPUT A$(N,16 TO 35)
125  PRINT A$(N,16 TO 35), "TOWN_ _ _ _ _ _";
130  INPUT A$(N,36 TO 55)
135  PRINT A$(N,36 TO 55),"COUNTY_ _ _ _";
140  INPUT A$(N,56 TO 67)
145  PRINT A$(N,56 TO 67), "POST_CODE_";
150  INPUT A$(N,68 TO 75)
155  PRINT A$(N,68 TO 75),"PHONE_ _ _ _ _";
160  INPUT A$(N,76 TO  )
165  PRINT A$(N,76 TO  )
170  LET N=N+1
175  IF INKEY$="" THEN GOTO 175
180  CLS
185  RETURN
300  CLS
305  PRINT AT 5,0;"NAME_TO_BE_DELETED?"
310  INPUT N$
```

```
315  FOR I=1 TO 5Ø
32Ø  IF N$<>A$(I, TO 15) THEN GOTO 345
325  FOR J=I TO 5Ø
33Ø  LET A$(J)=A$(J+1)
335  NEXT J
337  LET N=N-1
34Ø  GOTO 35Ø
345  NEXT I
35Ø  CLS
355  RETURN
5ØØ  CLS
5Ø5  PRINT AT 5,Ø;"NAME ENQUIRY?"
51Ø  INPUT N$
515  FOR I=1 TO 5Ø
52Ø  IF A$(I, TO 15)<>N$ THEN GOTO 535
525  PRINT ,,A$(I, TO 15),,A$(I,16 TO 35),A$(I,36 TO
     55),A$(I,56 TO 67),,A$(I,68 TO 75),,A$(I,76 TO 95)
527  LPRINT A$(I, TO 15),,A$(I,16 TO 35),A$(I,36 TO
     55), A$(I,56 TO 67),,A$(I,68 TO 75),,A$(I,76 TO 95)
53Ø  GOTO 54Ø
535  NEXT I
54Ø  IF INKEY$="" THEN GOTO 53Ø
545  CLS
55Ø  RETURN
7ØØ  FOR I=1 TO N-1
7Ø5  CLS
71Ø  PRINT AT 7,Ø;A$(I, TO 15),,A$(I,16 TO 35),A$(I,36
     TO 55),A$(I,56 TO 67),,A$(I,68 TO 75),,A$(I,76 TO
     95)
```

```
715  IF INKEY$="" THEN GOTO 715
720  NEXT I
725  CLS
730  RETURN
900  CLS
905  FOR I=1 TO N
910  PRINT AT 20,0;A$(I, TO 15)
915  SCROLL
920  NEXT I
925  CLS
930  RETURN
```

24

Resistencias eléctricas y colores

Este programa determina el valor de una resistencia a partir de su código compuesto por cuatro colores. El programa explica todo cuando se ejecuta.

Cuando pulse **RUN,** el programa le ofrecerá el siguiente menú:

"S" para volver al modo de comandos
"L" para aprender de qué se trata
"C" para conocer los códigos de los colores
"E" Ejemplos
"V" para convertir un valor al código de colores
"R" para convertir un código de colores a un valor
"H" para el menú

El programa se explica por sí mismo.

Programa 24: Código de resistencias

```
1 REM ********************************
      *  (C) MARK HARRISON           *
      *  - - - - - - - - - - - - - - -*
      *  RESISTOR CODE               *
      ********************************
2 REM
4 SLOW
```

```
5   LET R$="_ _ _ _ _ _ _ _▄▄▄▄▄▄▄▄▄
    ▄▄▄▄▄_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
    _ _ _▌_▒_▒_▒_ _ _ _▒_▐_ _ _ _
    _ _ _ _ _ _ _ _ _ _ _ _ _ _ _█ █▌_▒_▒_▒
    _ _ _ _▒_▐█ █_ _ _ _ _ _ _ _ _ _ _ _ _ _
    _ _ _ _▌_▒_▒_▒_ _ _ _▒_▐_ _ _
    _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _▀▀▀▀▀▀
    ▀▀▀▀▀▀▀▀_ _ _ _ _ _ _ _ _ _ _ _ _ _ _
    _ _ _ _ _ _ _ _1_2_3_ _ _ _4"
1Ø  LET S$="RESISTOR_CODE._ _ _ _ _ _ _ _ _ _
    _ _ _ _ _ _ _ _ _ _ _▀▀▀▀▀▀▀▀▀▀▀
    ▀▀▀"
15  LET A$="BLACK_BROWN_RED_ _ _
    ORANGEYELLOWGREEN_BLUE_ _
    VIOLETGREY_ _WHITE_"
2Ø  DIM C$(1Ø,6)
25  FOR I=1 TO 1Ø
3Ø  LET C$(I)=A$(I*6-5 TO )
35  NEXT I
4Ø  CLS
45  PRINT AT Ø,5;S$;AT 3,Ø;"ENTER:";AT 5,Ø;"""S""
    _TO_RETURN_TO_COMMAND_
    MODE,",""""L""_TO_LEARN_WHAT_IT_
    IS,",""""C""_FOR_COLOR_CODES,",""""E""_
    FOR_EXAMPLES,",
    """"V""_TO_CONVERT_VALUE_TO_
    RESISTOR_CODE,",,""""R""_TO_CONVERT_
    RESISTOR_CODE_TO_ _ _ _ _
    VALUE,",""""M""_FOR_MENU._"
```

```
 50  INPUT Q$
 55  IF Q$="S" THEN STOP
 60  IF Q$="C" THEN GOSUB 200
 65  IF Q$="E" THEN GOSUB 300
 70  IF Q$="V" THEN GOSUB 500
 75  IF Q$="R" THEN GOSUB 600
 80  IF Q$="L" THEN GOSUB 100
 85  GOTO 40
100  CLS
105  PRINT AT 0,5;S$;AT 2,5;"THE_VALUE_OF_
     RESISTORS_ARE_MARKED_BY_FOUR_
     COLORED_BANDS_ _ _WHICH_INDICATE_
     THEIR_RESISTANCE_VALUE_AND_THEIR_
     TOLERANCE."
110  PRINT AT 6,0;R$
115  PRINT AT 12,0;"_ _ _ _ _THE_FIRST_BAND_
     INDICATES_THE_FIRST_FIGURE_OF_THE_
     VALUE,_THE_SECOND_BAND_INDICATES_
     THE_SECOND_FIGURE_OF_THE_VALUE,_
     AND_THE_THIRD_BAND_INDICATES_HOW_
     MANY_ZEROS_THE_VALUE_CONTAINS."
120  PRINT AT 19,5;"THE_FOURTH_BAND_
     INDICATES_ _THE_TOLERANCE,_I.E._HOW_
     ACCURATEIT_IS."
125  INPUT Q$
130  GOSUB 200
135  IF Q$="M" THEN GOTO 150
140  GOSUB 300
150  GOTO 40
```

```
200  CLS
205  PRINT S$;AT 1,18;"COLOR_VALUE"
210  FOR I=1 TO 10
215  PRINT AT 1+2*I,18;C$(I);TAB 27;I-1
220  NEXT I
225  PRINT AT 5,0;"_ _ _THE_FIGURE",,"THAT_
     EACH_OF",,"THE_FIRST_THREE",,
     "COLORED_BANDS",,"REPRESENTS_
     IS",,"GIVEN_BY_THIS",,"TABLE."
230  PRINT AT 15,3;"THE_UNIT_OF",,
     "RESISTANCE_IS",,"OHMS."
235  INPUT Q$
240  IF Q$="M" THEN GOTO 270
250  CLS
255  PRINT S$;AT 4,4;"THE_TOLERANCE_OF_THE_
     _ _ _ _ _ _ _RESISTOR_IS_GIVEN_BY_
     THE_ _ _ _ _ _ _ _FOLLOWING_COLOR_
     VALUES_ _ _ _ _ _ _ _ _OF_THE_
     FOURTH_BAND:"
260  PRINT AT 10,0;"COLOR_ _TOL.
     (PERCENTAGE)",,,"BROWN_ _ _ _ _1",,
     "RED_ _ _ _ _ _ _2",,"GOLD_ _ _ _ _ _
     5",,"SILVER_ _ _10"
265  PRINT AT 17,3;"THE_TOLERANCE_BAND_IS_
     _ _ _ _ _ _ _OFTEN_ABSENT_FROM_THE_
     _ _ _ _ _ _ _ _ _ _RESISTOR._ _IN_THIS_
     CASE_ _ _ _ _ _ _ _ _ASSUME_A_20_
     PERCENT_ _ _ _ _ _ _ _ _ _ _ _ _
     TOLERANCE."
```

```
270  INPUT Q$
275  RETURN
300  CLS
305  PRINT AT 0,5;S$;AT
     3,0;"EXAMPLES.",,"----------"
310  PRINT AT 5,0;R$
315  PRINT AT 13,0;"BAND_1_ _BAND_ _2_ _
     BAND_3_ _BAND_4"
320  INPUT Q$
325  PRINT AT 15,0;"GREEN_ _ _BLACK_ _ _
     BLACK_ _ _GOLD"
330  PRINT AT 17,2;"50_X_1_=_50_
     OHMS.";AT 18,2;"5_PERCENT_
     TOLERANCE.";AT 20,0;"PRESS_ _
     ENTER_",,"OR_ENTER_""M""_FOR_
     MENU."
335  INPUT Q$
340  IF Q$="M" THEN GOTO 420
345  PRINT AT 15,0;"BLUE_ _ _ _GREEN_ _ _
     YELLOW_ _RED_ _ _"
350  PRINT AT 17,2;"65_X_10000_=_650,000_
     OHMS.";AT 18,2;"2_PERCENT_TOLERANCE."
355  INPUT Q$
360  IF Q$="M" THEN GOTO 420
365  PRINT AT 15,0;"GREY_ _ _ _VIOLET_ _
     GREEN_ _ _SILVER"
370  PRINT AT 17,2;"87_X_100000_=8,700,000_
     OHMS._";AT 18,2;"10_PERCENT_
     TOLERANCE._"
```

```
375  INPUT Q$
380  IF Q$="M" THEN GOTO 420
385  PRINT AT 15,0;"BLACK_ _ _BROWN_ _ _
     BLACK_ _ _BROWN_"
390  PRINT AT 17,2;"01_X_1_=_1_OHM._ _ _ _
     _ _ _ _ _ _ _ _ _ _";AT 18,2;"1_PERCENT_
     TOLERANCE._ _ _ _"
395  INPUT Q$
400  IF Q$="M" THEN GOTO 420
405  PRINT AT 15,0;"YELLOW_ _ORANGE_ _RED_
     _ _ _ _SILVER_"
410  PRINT AT 17,2;"43_X_100_=_4300_OHMS._
     _";AT 18,2;"10_PERCENT_TOLERANCE."
415  INPUT Q$
420  RETURN
500  CLS
505  PRINT AT 0,5;S$;AT 3,0;"TO_CONVERT_A_
     VALUE_TO_RESISTOR_ _COLOR_CODE."
510  PRINT AT 6,0;"ENTER_VALUE_OF_RESISTOR_
     (OHMS)"
515  INPUT V$
520  IF LEN V$<1 OR LEN V$>11 THEN GOTO 515
525  PRINT AT 8,0;R$;AT 15,4;V$;"_
     OHMS.";AT 17,0;"_ _ _ _BAND_1_ _BAND_
     2_ _BAND_3"
530  IF LEN V$=1 THEN LET V$="0"+V$
535  PRINT "_ _ _ _";C$(VAL V$(1)+1);"_ _
     ";C$(VAL V$(2)+1);"_ _";C$(LEN V$-1)
540  PRINT AT 21,4;"ENTER_""M""_FOR_MENU."
```

25

Resultados del examen

Este programa permite que se introduzcan los nombres de los alumnos y sus resultados en un examen. Después, el programa imprime los nombres de los alumnos en el orden de sus resultados, junto con el resultado expresado como un porcentaje y como un grado.

Introduzca primero el nombre del alumno y después el resultado del examen. Al introducir el último resultado, pulse **STOP** (shift-A). Si se comete un error, escriba "N" cuando aparezca el mensaje CORRECTO? en la pantalla. De este modo podrá volver a introducir la información. Los resultados pueden guardarse en cinta, en cuyo caso cuando la cinta se carga son accesibles de inmediato.

Programa 25: Resultados del examen

```
 5 SLOW
10 LET G$="A+A_A-B+B_B-C+C_C-D+D
   D-E+E_E-"
15 DIM N$(150,22)
20 DIM M(150)
25 PRINT AT 0,5;"EXAMINATION_GRADER.";AT
   1,5;"■■■■■■■■■■■■■■■■■■■■
   ■■";AT 2,5;"THIS_PROGRAM_WILL_
   ALLOW_ _ _ _PUPILS_NAMES_AND_
   THEIR_EXAM_ _ _ _ _RESULTS_TO_BE_
```

```
    ENTERED_AND_THEN_ _PRINTS_OUT_
    THE_PUPILS_NAMES_IN_ _THE_ORDER_
    OF_THEIR_RESULTS,_WITHTHEIR_RESULT_
    WRITTEN_AS_A_ _ _ _ _ _ _ _
    PERCENTAGE_AND_A_GRADE."
30  PRINT AT 9,5;"ENTER_FIRST_THE_NAME_
    OF_THEPUPIL_AND_THEN_THEIR_
    RESULT_IN_ _THE_EXAM._ _AFTER_THE_
    LAST_RESULTENTER_""_STOP_""_(SHIFT_
    A)._ _IF_ANERROR_IS_MADE,_ENTER_
    ""N""_WHEN_ _ _""CORRECT?""_
    APPEARS."
35  PRINT AT 15,5;"RESULTS_MAY_BE_
    ""SAVED""_ON_ _TAPE_AND_WHEN_
    ""RE-LOADED""_CAN_BEOBTAINED_BY_
    CALLING_FOR_THE_MENUIMMEDIATELY."
40  PRINT AT 19,0;"ENTER:";AT 20,0;"""""""_ _
    TO_INPUT_NAMES_AND_
    RESULTS","""M""_FOR_MENU"
45  INPUT Q$
50  IF CODE Q$=50 THEN GOTO 260
55  CLS
60  PRINT AT 10,0;"WHICH_GRADING_
    SYSTEM";AT 12,0;"ENTER:",,"""A""_FOR_ _
    A,B,C",,"""B""_FOR_ _A,B,C,D,E"
65  INPUT Q$
67  LET G=9+(6 AND Q$="B")
70  CLS
```

```
 75  PRINT AT 1Ø,Ø;"ENTER_MAXIMUM_RESULT_
     IN_EXAM"
 8Ø  INPUT M$
 85  PRINT AT 12,1Ø;M$;AT 14,8;"CORRECT?"
 9Ø  INPUT Q$
 95  IF CODE Q$=51 THEN GOTO 7Ø
1ØØ  LET M=INT ABS VAL M$
1Ø5  FOR I=1 TO 15Ø
11Ø  CLS
115  PRINT AT 9,Ø;"ENTER_NAME_OF_PUPIL_";I
12Ø  INPUT N$(I)
125  IF CODE N$(I)=227 THEN GOTO 17Ø
13Ø  PRINT AT 11,Ø;N$(I);AT 14,Ø;"ENTER_PUPILS_
     EXAM_RESULT";
135  INPUT R$
14Ø  PRINT AT 16,Ø;R$;AT 19,Ø;"CORRECT?"
145  INPUT Q$
15Ø  IF CODE Q$=51 THEN GOTO 11Ø
155  LET M(I)=INT (VAL R$*1ØØ/M)
16Ø  LET N=I
165  NEXT I
17Ø  FAST
175  FOR J=2 TO N
18Ø  LET T$=N$(J)
185  LET T=M(J)
19Ø  FOR I=J-1 TO 1 STEP -1
195  IF M(I)>T THEN GOTO 215
2ØØ  LET N$(I+1)=N$(I)
2Ø5  LET M(I+1)=M(I)
```

```
210  NEXT I
215  LET N$(I+1)=T$
220  LET M(I+1)=T
225  NEXT J
230  LET N$(1,21 TO 22)="A+"
235  LET K=1
240  FOR I=2 TO N
245  IF I>K*N/G AND M(I)<>M(I-1) THEN LET
     K=K+1
250  LET N$(I,21 TO 22)=G$(2*K-1 TO 2*K)
255  NEXT I
260  CLS
265  SLOW
270  PRINT AT 4,0;"PRESS";AT 6,0;""""C""" _FOR_
     PRINTED_COPY";AT 7,0;""""D""" _TO_
     DISPLAY_RESULTS";AT 8,0;""""M""" _TO_
     RETURN_TO_COMMAND_MODE";AT 9,0;
     """"R""" _TO_RE-RUN_PROGRAM";
     AT 10,0;""""S""" _TO_SAVE_RESULTS"
275  PAUSE 40000
280  LET Q$=INKEY$
285  CLS
290  IF Q$="M" THEN STOP
295  IF Q$="R" THEN RUN
300  IF Q$="S" THEN SAVE "EXAM"
305  IF Q$<>"D" THEN GOTO 365
310  FOR I=1 TO N
315  PRINT I;TAB 3;N$(I);TAB 27;M(I)
320  IF I<>15*INT (I/15) THEN GOTO 345
```

```
325  PRINT AT 18,Ø;"PRESS_ENTER_ _ _";AT
     2Ø,Ø;"OR_ENTER_ANY_CHARACTER_TO_
     RETURNTO_MENU"
33Ø  INPUT Q$
335  IF Q$<>"" THEN GOTO 26Ø
34Ø  CLS
345  NEXT I
35Ø  PRINT ,, TAB 7;"**_ _**_ _**_ _**"
355  INPUT Q$
36Ø  GOTO 26Ø
365  IF Q$<>"C" THEN GOTO 26Ø
37Ø  FOR I=1 TO N
375  LPRINT I;TAB 3;N$(I);TAB 27;M(I)
38Ø  NEXT I
385  GOTO 26Ø
9999 FAST
```

APENDICES

1

Para acelerar la ejecución de programas

Gran parte de los métodos que se utilizan para ahorrar espacio en la computadora T/S 1000 desaceleran la ejecución de los programas. En ocasiones es preferible ocupar más espacio, con tal de incrementar la velocidad de ejecución.

En este capítulo se verán algunos programas cuya diferencia en velocidad de ejecución puede parecer marginal si forman parte de un programa corto. Sin embargo, con programas largos o con rutinas que se repiten muchas veces dentro de un bucle, la diferencia entre dos programas, uno escrito intentando optimizar la velocidad y otro no, puede ser considerable. Las sugerencias de este capítulo se refieren a programas escritos para ejecutarse en el modo lento **(SLOW).** En general, el modo rápido no requiere estas atenciones, excepto en ocasiones especiales.

Estructura

Los más importante es que el programa esté bien estructurado. Deben ponerse las partes del programa que en más ocasiones se ejecuten al principio del listado y las partes menos frecuentemente se utilicen al final del mismo. Hay una justificación muy simple para este proceder. Las instrucciones como **GOTO** y **GOSUB** tienen que buscar a lo largo del programa hasta encontrar el número de línea al que transfieren el control. Por lo tanto, las subrutinas y bucles del programa se eje-

cutarán más rápidamente si están al principio del programa que si estuvieran al final. Para que el programa comience utilizando la sentencia **PRINT** para escribir los literales que indiquen al usuario lo que hace el programa y cómo debe usarle, se puede arbitrar un truco. Por ejemplo, puede comenzarse el programa con una instrucción **GOSUB** 9000 y colocar a partir de esta línea los mensajes al usuario que se deseen.

Sentencias PRINT

Escriba los caracteres directamente entre comillas siempre que se pueda. Compruebe el tiempo de ejecución de estos programas:

```
10 FOR F = 1 TO 350
20 PRINT "0"; "0";
30 NEXT F
```

```
10 FOR F = 1 TO 350
20 PRINT 0;0;
30 NEXT F
```

```
10 FOR F = 1 TO 350
20 PRINT CHR$ 28; CHR$ 28;
30 NEXT F
```

Tardarán aproximadamente nueve, once y trece segundos, respectivamente. La diferencia es tan aparente que no se necesita reloj para cronometrar.

Unión de sentencias PRINT

El usuario debe saber que en una línea de un programa se admite cualquier combinación de las instrucciones **PRINT/TAB/AT.** Sin embargo, debe tenerse en cuenta que los programas ganan velocidad si las sentencias **PRINT** se encadenan que si fueran una serie de líneas de programa. Para corroborar esto, ejecute y cronometre los siguientes programas:

```
1Ø  FOR N=1 TO 1ØØ
2Ø  PRINT AT 1Ø,11;"THINKING"
3Ø  PRINT AT 1Ø,11;"THINKING"
4Ø  PRINT AT 1Ø,11;"THINKING"
5Ø  PRINT AT 1Ø,11;"THINKING"
6Ø  NEXT N
```

```
1Ø  FOR N=1 TO 1ØØ
2Ø  PRINT AT 1Ø,11;"THINKING";AT 1Ø,11;
    "THINKING";AT 1Ø,11;
    "THINKING";AT 1Ø,11;"THINKING"
3Ø  NEXT N
```

Operadores lógicos y relacionales

Estas instrucciones son de evaluación lenta. Cualquier expresión que dé como resultado un cero (falso) se computa antes que otra que dé como resultado uno (verdadero). Es decir, las sentencias falsas se ejecutan más rápidamente que las verdaderas. Esto no es todo. Una expresión como **IF** X **THEN ...** es más rápida que **IF** X = 1 **THEN ...** y **IF NOT** X **THEN ...** es más rápido que **IF** X = Ø **THEN ...**

Observará que **IF** X <> 1 **THEN ...** se evalúa ligeramente más rápido que **IF NOT** X = 1 **THEN ...,** aunque la diferencia es inapreciable.

Trate de escribir sus sentencias condicionales de tal modo que en la mayor parte de los casos sean falsas. Obviamente, si para conseguir este efecto tiene que introducir otros elementos que desaceleran el programa, el esfuerzo se habrá hecho en vano.

Almacenamiento de pantallas en cinta

Si se tiene un programa que utilice gráficos complicados, tales como curvas sinusoidales o elipses, puede merecer la pena

ejecutar el programa para dibujar los gráficos, para después almacenar los gráficos en cinta. Una rutina que realice complejos dibujos se ejecutará lentamente, mientras que cargar un dibujo desde la cinta magnética es instantáneo.

2

Caracteres expandidos

La memoria ROM de la computadora T/S mantiene una tabla de datos para generar los caracteres de la pantalla. Esta tabla se denomina generador de caracteres y empieza en la dirección 7680 (decimal) y acaba en la dirección 8191 (decimal). La tabla contiene una serie de ceros y unos que corresponden a puntos en blanco o en negro en la pantalla. Cada carácter comprendido entre el 0 y 63 se componen de un cuadrado de bits 8 × 8, los demás caracteres son combinación de éstos. El generador de caracteres contiene ocho octetos por cada carácter. Vamos a ver un ejemplo con el número 0. Este número tiene como código (**CODE**) 28 y su definición empieza en la dirección 7904. La forma de averiguar donde empieza la definición de un carácter es muy simple:

7680 +(**CODE** * 8)

y termina siete octetos más adelante

7680 +(**CODE** * 8 + 7)

Si se examinan los octetos de memoria correspondientes al 0 se obtiene:

```
7904  00000000
7905  00111100
7906  01000110
7907  01001010
7908  01010010
7909  01100010
7910  00111100
7911  00000000
```

Si se mira esta figura como si los ceros no existieran se observará algo como esto:

```
7904
7905     1111
7906   1     11
7907   1   1 1
7908   1 1   1
7909   11    1
7910    1111
7911
```

que es similar a como aparece en la pantalla. La manera de averiguar que bits tiene uno y cuáles cero es dividir repetidamente por dos y el resto nos va indicando los valores de los bits.

A continuación se presenta un programa que utiliza la función **PLOT** para crear caracteres cuyo tamaño es cuatro veces mayor que el normal. Cuando ejecute este programa, introduzca un carácter cada vez cuando aparezcan en pantalla las comillas correspondientes al **INPUT,** después pulse ENTER. Los caracteres se dibujan en la esquina superior izquierda de la pantalla. El programa ocupa 1K de memoria.

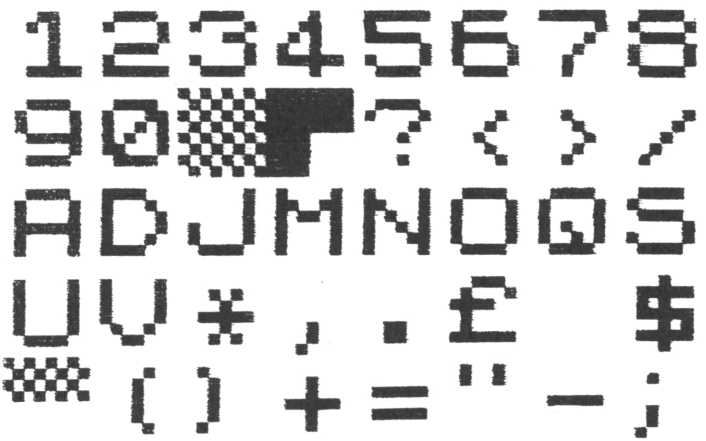
Caracteres expandidos

```
30 INPUT A$
40 CLS
50 FOR A=0 TO 7
60 LET P=PEEK (7680+8*CODE A$+A)
70 FOR B=0 TO 7
80 IF P-2*INT (P/2) THEN PLOT 7-B,43-A
90 LET P=INT (P/2)
```

```
100 NEXT B
110 NEXT A
120 GOTO 30
```

Esta versión del programa sólo permite tener un carácter expandido en la pantalla cada vez. De este modo el programa es muy simple. La siguiente versión del programa mediante las variables DOWN y ACROSS controla dónde se imprime cada carácter en la pantalla. Esta versión sólo permite caracteres cuyo código esté comprendido entre 0 y 63.

Caracteres expandidos utilizando PLOT

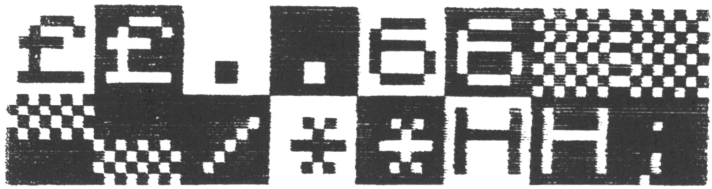


```
10 LET ACROSS=7
20 LET DOWN=43
30 INPUT A$
40 LET C=CODE A$
50 FOR K=0 TO 7
60 LET P=PEEK (7680+C*8+K)
70 FOR F=0 TO 7
```

```
 8Ø  IF P-2*INT (P/2) THEN PLOT
     ACROSS-F,DOWN-K
 9Ø  LET P=INT (P/2)
1ØØ  NEXT F
11Ø  NEXT K
12Ø  LET ACROSS=ACROSS+8
13Ø  IF ACROSS>63 THEN LET DOWN=DOWN-8
135  IF ACROSS>63 THEN LET ACROSS=7
14Ø  GOTO 3Ø
```

Caracteres expandidos en negativo utilizando PLOT

Esta versión permite dibujar para los caracteres cuyo código está comprendido entre Ø y 63 sus formas expandidas en positivo y negativo. Todas las palabras clave, funciones, etc., son combinaciones de estos caracteres. En la pantalla caben cinco filas de ocho caracteres expandidos cada una, es decir, un total de 40 caracteres. Esta es una de las ventajas de usar **PLOT** en lugar de **PRINT AT.** Quiere decirse que aunque con **PLOT** los caracteres obtenidos son más pequeños, caben más en la pantalla.



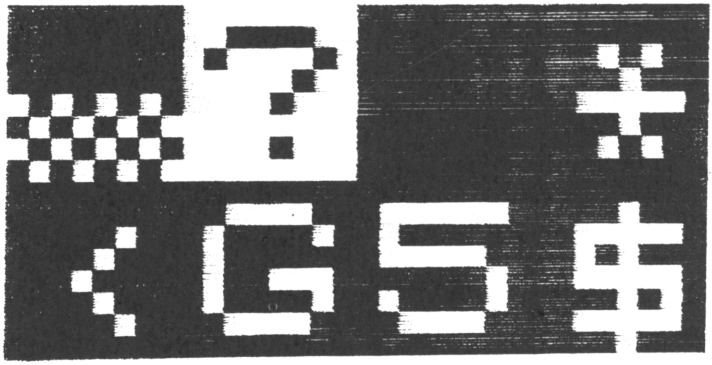
```
1Ø  LET ACROSS=7
2Ø  LET DOWN=43
3Ø  INPUT A$
```

```
40  LET C=CODE A$
45  LET CC=C-(128 AND C>128)
50  FOR K=0 TO 7
60  LET P=PEEK (7680+CC*8+K)
70  FOR F=0 TO 7
80  IF C>=128 AND P-2*INT (P/2)=0 THEN PLOT
    ACROSS-F,DOWN-K
85  IF C<128 AND P-2*INT (P/2)<>0 THEN PLOT
    ACROSS-F,DOWN-K
90  LET P=INT (P/2)
100 NEXT F
110 NEXT K
120 LET ACROSS=ACROSS+8
130 IF ACROSS>63 THEN LET DOWN=DOWN-8
135 IF ACROSS>63 THEN LET ACROSS=7
140 GOTO 30
```

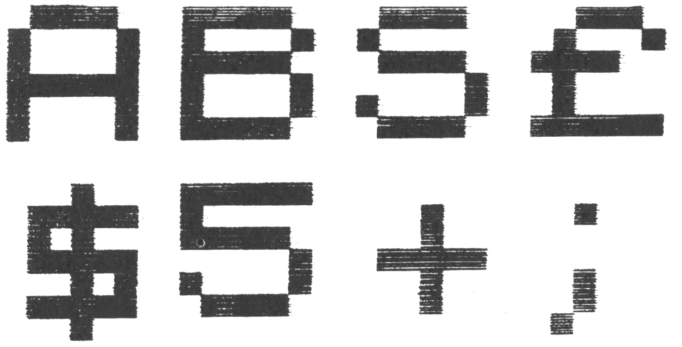
En este programa se ha introducido una variable adicional CC. C es el código **(CODE)** del carácter que se va a expandir y CC es el código reducido a un número comprendido entre 0 y 63 para que se pueda examinar el generador de caracteres. Si C es mayor que 127 se le convierte a su negativo en las líneas 80 y 85 para determinar que areas son blancas y cuáles negras.

Caracteres expandidos utilizando PRINT AT

Ahora vamos a utilizar **PRINT AT.** Aunque este procedimiento tiene las desventajas ya comentadas, es posible conseguir caracteres de tamaño superior a los que se consiguen con **PLOT** y permite elegir entre caracteres negros sobre fondo blanco o caracteres blancos sobre fondo negro. Un aspecto a mencionar es que la coordenada Y para **PLOT** y **PRINT AT** funcionan de forma opuesta.



```
10  LET ACROSS=0
20  LET DOWN=0
30  INPUT A$
40  LET C=CODE A$
45  LET CC=C-(128 AND C>=128)
50  FOR K=0 TO 7
60  LET P=PEEK (7680+CC*8+K)
70  FOR F=0 TO 7
80  IF P-2*INT (P/2)<>0 AND C<128 THEN PRINT
    AT DOWN+K,ACROSS+7-F;"■"
90  IF P-2*INT (P/2)=0 AND C>=128 THEN PRINT
    AT DOWN+K,ACROSS+7-F;"■"
100 LET P=INT (P/2)
110 NEXT F
120 NEXT K
130 IF ACROSS+8>31 THEN LET DOWN=DOWN+8
140 LET ACROSS=ACROSS+8 AND
    ACROSS+8<=31
150 GOTO 30
```

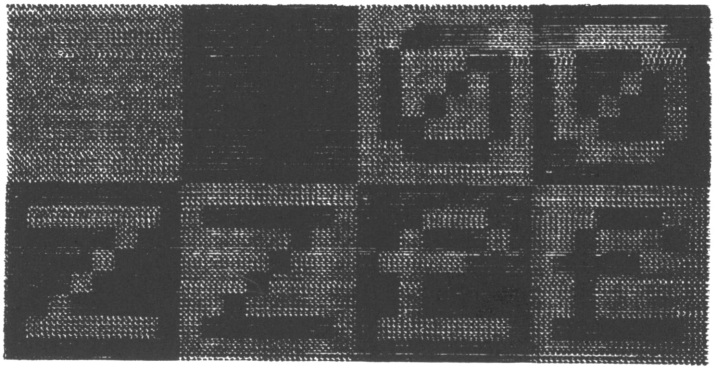


Caracteres expandidos con diferentes tonalidades

Por último una versión que permite imprimir en negro sobre gris o viceversa (en los caracteres en negativo). No funciona muy bien en la impresora pero sí en la pantalla. Para utilizar diferentes caracteres debe cambiar las líneas 80 y 90. Intente escribir una versión que le permita elegir los caracteres para los caracteres y el fondo. (Pista: utilice una cadena.)

```
10  LET ACROSS=0
20  LET DOWN=0
30  INPUT A$
40  LET C=CODE A$
45  LET CC=C-(128 AND C>=128)
50  FOR K=0 TO 7
60  LET P=PEEK (7680+CC*8+K)
70  FOR F=0 TO 7
75  LET PP=P-2*INT (P/2)
80  IF (PP<>0 AND C<128) OR (PP=0 AND
    C>=128) THEN PRINT AT DOWN+K,
    ACROSS+7-F;"■"
```

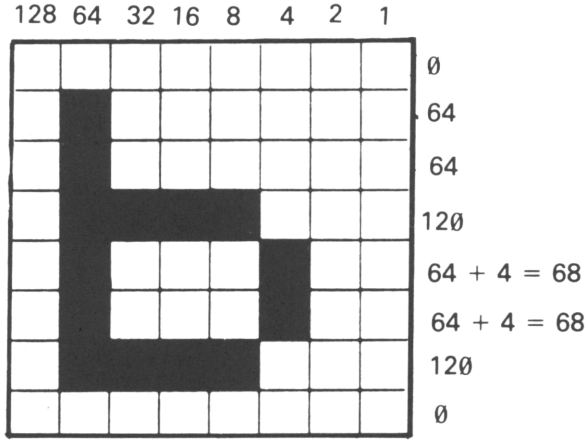
```
90  IF (PP=Ø AND C<128) OR (PP<>Ø AND
    C>=128) THEN PRINT AT DOWN+K,
    ACROSS+7-F;"▩"
100 LET P=INT (P/2)
110 NEXT F
120 NEXT K
130 IF ACROSS+8>31 THEN LET DOWN=DOWN+8
140 LET ACROSS=ACROSS+8 AND
    ACROSS+8<=31
150 GOTO 3Ø
```



Recuerde que no tiene por qué limitarse a aquellos caracteres cuya definición figura ya en la memoria ROM. Mientras que tengan el mismo formato (ocho octetos por carácter) y conozca dónde empieza puede diseñar un alfabeto de letras minúsculas y almacenarlo mediante una sentencia **REM** al principio del programa, después puede utilizarse la función **PEK** para encontrar el dato preciso. Piense lo satisfactorio que puede resultar mostrar a sus amigos que su computadora T/S 1000 dispone de letras minúsculas.

Se dará cuenta de que un trozo de papel cuadriculado es imprescindible. Para diseñar cualquier carácter utilice un cua-

drado 8 × 8 como se muestra más abajo. Rellene aquellos cuadros que le apetezca. Para cada fila sume los números que figuran arriba de las zonas sombreadas, de esta forma obtendrá un número por cada fila y tendrá por tanto 8 números. Examine el ejemplo.



Ahora dispone de una secuencia de números.

Ahora utilice la función **POKE** para almacenar estos ocho números en una sentencia **REM.** Se ha simplificado bastante con objeto tan sólo de clarificar. Estudie los ejemplos anteriores antes de ampliar esta rutina.

```
 1  REM RNDRND????
1Ø  FOR A=Ø TO 7
2Ø  LET P=PEEK (16514+A)
3Ø  FOR B=Ø TO 7
```

```
40 IF P-2*INT (P/2) THEN PLOT 7-B,43-A
50 LET P=INT (P/2)
60 NEXT B
70 NEXT A
```

Este programa le proporcionará una b minúscula cuyo tamaño es cuatro veces el normal. Experimente con el programa e intente ampliarle para incluir más caracteres, utilizando ejemplos anteriores conseguirá alguna ayuda.

3

Variables del sistema útiles

Las variables del sistema están almacenadas en memoria RAM entre las direcciones 16384 y 16508. Indican a la computadora cosas sobre sí misma, tales como, dónde empieza el fichero de representación visual o donde empieza el área destinada al almacenamiento del programa o de las variables en la memoria. Esta información puede utilizarse en los programas e incluso puede alterarse el valor de algunas de estas variables del sistema para permitir así a la computadora realizar funciones especiales como cronometrar. Por medio de la función **PEEK** puede leerse el valor de cualquiera de estas variables del sistema, aunque no todas le serán de utilidad al programar.

Dirección	Comentario
16384	Controla los mensajes de error. Si se almacena uno de los siguientes valores en este lugar, el programa se detiene sin que aparezca el correspondiente mensaje de error: 43, 70, 72, 73, 74, 75, 76, 77, 79, 81, 82 ó 89. El sistema está ahora en condiciones de interpretar cualquier tecla que se pulse como una orden.
16388/16389	Contiene la dirección del primer octeto que sigue al área destinada al programa BASIC, algunas veces se denomina a esta variable del sistema RAMTOP. La instruc-

Dirección	Comentario
	ción **NEW** sólo limpia las posiciones de memoria anteriores a la dirección contenida en RAMTOP, por tanto cualquier información almacenada en direcciones superiores permanece intacta. Si el valor de esta variable es superior a 19712 entonces cuando se ejecuta **CLS,** el fichero de representación visual se contrae hasta su mínima extensión.
16396/16397	Contiene la dirección del principio del fichero de representación visual. No debe alterarse el valor de esta variable. Del mismo modo esta variable indica la extensión del programa (exceptuando variables, pantalla, etc.
16398/16399	Almacena la posición en la que escribirá la próxima sentencia **PRINT** en el fichero de representación visual. Por tanto modificándola adecuadamente se puede escribir en cualquier lugar de la pantalla.
16400/16401	Contiene la dirección del principio del área de memoria destinada a las variables. Es útil cuando se programa en lenguaje máquina.
16419/16420	Almacena el número de la mayor línea de programa que aparecerá al listarle. Puede cambiarse el valor de modo que no se pueda listar todo el programa.
16425	Almacena la dirección de la siguiente línea que será ejecutada. Es útil para modificar líneas de un programa desde dentro del propio programa.
16434	Contiene el origen para la función **RND.** Es útil en programas que generen números o caracteres aleatorios utilizando las funciones **RND/RAND,** puesto que esta es la variable que fija la función **RAND.**

Dirección	Comentario
16436/16437	Contiene el contador de imágenes. El bit 15 es 1. Los bit desde el 0 al 14 se decrementan de uno en uno por cada imagen que se envía a la TV. El número total de imágenes es de 32768. La función **PAUSE** sitúa un 0 en el bit 15 y en los otros bit pone la duración de la pausa. La función **PAUSE** concluye cuando se alcanza 0 y en ese momento el bit 15 vuelve a valer uno.
16438	Contiene la coordenada X del último pixel en el que se haya dibujado mediante la función **PLOT.**
16439	Contiene la coordenada Y del último pixel en el que se haya dibujado mediante la función **PLOT.**
6441	Almacena el número de la columna de la posición de **PRINT** en la pantalla.
16442	Almacena el número de línea de la posición de **PRINT** en la pantalla.
16444 a 16476	Contiene el almacenamiento temporal de datos de la impresora. Son 32 caracteres de una línea más ENTER. Si no se está utilizando impresora puede utilizarse para almacenar temporalmente otros datos.

4

Convertidor de decimal a hexadecimal

Este programa tan corto convierte cualquier número decimal comprendido entre 0 y 255 a su equivalente hexadecimal. Es útil para quienes tengan intención de programar en lenguaje máquina.

```
1Ø  PRINT AT Ø,Ø;"ENTER_A_DECIMAL_NO. (Ø_
    TO_255)"
2Ø  INPUT X
3Ø  SCROLL
4Ø  IF X<>INT X OR X<Ø OR X>255 THEN GOTO 1Ø
5Ø  PRINT TAB 3-LEN STR$ X;X;"_DECIMAL_IS_";
6Ø  PRINT CHR$ (28+INT (X/16)) + CHR$ (28+(X-INT
    (X/16)*16));
7Ø  PRINT "_HEXADECIMAL"
8Ø  GOTO 1Ø
```

La línea 10 pregunta al usuario el número que desea convertir, utilizando un mensaje que aparece en la parte superior de la pantalla, de tal forma que cuando la pantalla se "corre" hacia arriba el mensaje desaparece. La línea 40 rechaza los valores que están fuera del rango o que no son enteros y solicita del usuario una nueva entrada. La línea 50 manipula los espa-

cios de manera adecuada. Para ello examina la longitud del número convertido a cadena mediante la función **STR$.** La línea 60 realiza la conversión. Puede ahorrarse memoria si se combinan las instrucciones **PRINT** de las líneas 50, 60 y 70 en una sola sentencia.

5

Convertidor de hexadecimal a decimal

Este programa que ocupa 1K convierte cualquier valor hexadecimal comprendido entre ØØ y FF (un octeto) a su equivalente decimal, es decir, este programa es la función inversa del precedente. Tal vez le interese juntar ambos programas y añadir la posibilidad de elegir entre las dos opciones.

```
1Ø  PRINT AT Ø,Ø;"ENTER_A_HEXADECIMAL_
    NO.ØØ_TO_FF"
2Ø  DIM X$(2)
3Ø  INPUT X$
4Ø  SCROLL
5Ø  IF X$(1)<"Ø" OR X$(1)>"F" OR X$(2)<"Ø" OR
    X$(2)>"F" THEN GOTO 1Ø
6Ø  PRINT X$;"_HEX_IS_";
7Ø  LET X=(CODE X$-28)*16+CODE X$(2)-28
8Ø  PRINT TAB 14-LEN STR$ X;X;"_DECIMAL"
9Ø  GOTO 1Ø
```

El programa es muy similar al anterior; los valores hexadecimales en el rango en el que se está interesado tienen como máximo dos caracteres. Por esto, la línea 20 asegura que X$ (la cadena hexadecimal a convertir) contiene siempre

2 caracteres. La línea 50 comprueba que los caracteres de X$ son permisibles, es decir, son dígitos hexadecimales comprendidos entre 0 y F. La línea 60 escribe la primera parte de la conversión, en esta ocasión no se precisa ningún método especial para controlar el espaciado, puesto que siempre los valores hexadecimales tienen la misma longitud. La línea 70 convierte el valor hexadecimal a un número decimal comprendido entre 0 y 255. Ahora sí que se requiere tener en cuenta los espacios y esta labor se realiza de manera similar al anterior programa.

6

Símbolos gráficos especiales

Los símbolos gráficos de la siguiente lista están disponibles en la computadora T/S. Apréciese que en la lista cada símbolo aparece enmarcado, es decir, que sus bordes están dibujados aunque no vayan a aparecer en la pantalla. Por ejemplo, el primer símbolo de la lista en la pantalla aparecerá exclusivamente como un espacio en lugar de como un cuadrado sin rellenar como aparece en la lista. Debe tenerse también en cuenta que mediante el modo G se accede a los caracteres en negativo.

Símbolo	Modo	Tecla	Símbolo	Modo	Tecla
	K or L	SPACE		G	SPACE
	G	shifted-1		G	shifted-Q
	G	shifted-2		G	shifted-W
	G	shifted-7		G	shifted-6
	G	shifted-4		G	shifted-R
	G	shifted-5		G	shifted-8
	G	shifted-T		G	shifted-Y
	G	shifted-E		G	shifted-3
	G	shifted-A		G	shifted-H
	G	shifted-D		G	shifted-G
	G	shifted-S		G	shifted-F

OTRAS OBRAS DE INTERES
PUBLICADAS POR OSBORNE/McGRAW-HILL

ADAM OSBORNE: Guía del comprador de sistemas de gestión
ADAM OSBORNE: Guía del ordenador personal PET/CBM
ANNIE FOX: BASIC básico. Guía para principiantes
CASTLEWITZ: Introducción al Visicalc
JOHN HEILBORN: Programas para ciencias e ingeniería. Edición APPLE II
JOHN HEILBORN: VIC 20. Guía del usuario
LON POOLE: Algunos programas de uso común en BASIC
LON POOLE: APPLE II. Guía del usuario
LON POOLE: Algunos programas de uso común en PASCAL
LON POOLE: Programas prácticos en BASIC. Edición APPLE II
LON POOLE: Programas prácticos en PASCAL
LON POOLE: Algunos programas de uso común en BASIC. Edición IBM
LON POOLE: Algunos programas de uso común en BASIC. Edición PET/CBM
LON POOLE: Algunos programas de uso común en BASIC. Edición TRS-80
LON POOLE: Algunos programas de uso común en BASIC. Edición APPLE II
LON POOLE: Algunos programas de uso común en BASIC. Edición ATARI
LON POOLE: Programas prácticos en BASIC
LON POOLE: Programas prácticos en BASIC. Edición TRS-80
LON POOLE: Programas prácticos en BASIC. Edición IBM
LYLE J. GRAHAM: IBM/PC. Guía del usuario
ROBERT MOTTOLA: Programación en lenguaje ensamblador para el APPLE II
TOM HOGAN: Sistema operativo CP/M. Guía del usuario. 2.ª ed.
WALTER ETTLIN: Introducción al WORDSTAR
MITCHELL WAITE: Introducción al procesamiento de palabras

DISCOGUIAS PUBLICADOS POR OSBORNE/McGRAW-HILL

CURTIS A. INGRAHAM: Discoguía para CP/M
DAVID A. WILSON: Discoguía para IBM/PC
DAVID A. WILSON: Discoguía para VISICALC
JOHN TAYLOR: Discoguía para ATARI 400/800
ZELDA GIFFORD: Discoguía para APPLE II

OTRAS OBRAS DE INTERES
PUBLICADAS POR BYTE-BOOKS/McGRAW-HILL

ABELSON: APPLE LOGO
BOWLES: Introducción al UCSD Pascal
CIARCIA: Construya una computadora basado en el Z-80 (Guía de diseño y funcionamiento)
KAMINS: Usted y la microcomputadora (Una introducción humanizada a la microinformática)
LEWART: Programas de ciencias e ingeniería para microcomputadoras SINCLAIR ZX81 COMPATIBLES CON EL ZX SPECTRUM
MORGAN: Introducción al microprocesador 8086/8088 (16 bit)
PECKAM: BASIC para IBM. Manual Práctico
PECKAM: BASIC para TRS-80 color. Manual Práctico
PECKHAM: BASIC para APPLE II. Manual Práctico
SIKONOWIZ: Introducción al IBM/PC
WATT: Aprendiendo con LOGO

ISBN: 968-451-705-X