5

Escrever bons programas é uma arte que se pode aprender, e o propósito desta obra é ensiná-lo por meio de exemplos escolhidos.

Os vários programas são fonte de referência de técnicas muito difundidas no mundo da informática: alguns deles ilustram, por exemplo, o uso de registo de comprimento fixo ou de comprimento variável, a sua ordenação, a pesquisa binária; outros apresentam técnicas especialmente adaptadas ao Spectrum, **atribuindo-se especial importância ao uso adequado da imagem.**

Andrew Hewson escreve uma coluna mensal na revista Sinclair User, **da Sinclair, respondendo a perguntas escritas relativas ao** hardware **e ao** software **dos computadores** ZX. **É também autor de várias obras sobre o mesmo tema.**

Todos os programas deste livro foram verificados e testados pelo Gabinete Verbo de Informática.

BIBLIOTECA VERBO DE INFORMÁTICA

Verbo OS 20 MELHORES PROGRAMAS A. HEWSON

OS 20 MELHORES PROGRAMAS

PARA O ZX SPECTRUM

ANDREW HEWSON

VERBO

BIBLIOTECA DE INFORMÁTICA

Os 20 Melhores Programas para o ZX SPECTRUM

ANDREW HEWSON

Os 20 Melhores Programas

para o ZX SPECTRUM

Verbo

Título do original inglês:
20 Best Programs for the ZX Spectrum
Tradução e revisão dos programas
de Carlos Sebastião e Silva

Editorial Verbo, S. A. R. L. — Lisboa/São Paulo
N.º Ed. 1622
Composto por Fotocompográfica
Impresso por Litográfica do Sul
em Junho de 1985
Dep. Legal n.º 9038/85

ÍNDICE

Nenhum livro desta natureza é um trabalho puramente individual. Agradeço ao meu irmão Gordon toda a sua lealdade e apoio. Os meus agradecimentos vão também para Louise Esplin, que decifrou a minha letra e dactilografou o manuscrito. Agradeço principalmente a Janet por ter tomado tanta coisa a seu cargo, de modo a permitir-me ter a liberdade suficiente para concluir o meu trabalho.

A D H

Introdução

Em 1980, a Sinclair Research Ltd lançou no mercado o primeiro computador a preço inferior a 100 libras: o *ZX80*. Possuía apenas 1 k (unidade de capacidade de memória) de RAM (memória de acesso directo), operava só com números inteiros e dispunha de capacidades de comando de imagem tão limitadas que a imagem desaparecia durante os cálculos; o *ZX80* era um parente pobre dos outros microcomputadores então disponíveis no mercado. Contudo, como o preço se situava a cerca de um quarto do dos outros minicomputadores, o *ZX80* ganhou rapidamente adeptos e, em breve, era o computador mais vendido do Reino Unido.

Na Primavera de 1981, a Sinclair lançou um substituto, o *ZX81*. Este computador ainda possuía apenas 1 k de memória RAM, mas operava já com números não inteiros e, se o desejássemos, mantinha permanentemente a imagem, embora pelo preço de uma enorme redução na velocidade de cálculo. O preço do *ZX81* situava-se ligeiramente abaixo das setenta libras.

A Sinclair lança então o seu computador de 1982, o *ZX Spectrum*. Possui, na sua versão base, 16 k de memória RAM interna, gera imagens a cores com uma resolução (ou definição) máxima de 256×176 *pixels* (pontos constituintes da imagem no *écran*) e dispõe de muitos melhoramentos de *software* em relação ao *ZX81*, incluindo possibilidades de controle de *diskettes*. Custava, na altura, 125 libras.

As capacidades do *ZX80* e, mais particularmente, do *ZX81* não são de desprezar, embora se reconheça que se destinam principalente a utilizadores neófitos que querem explorar o mundo da computação o mais economicamente possível. É necessária alguma habilidade para obter destas máquinas os melhores resultados — e isso não é um factor negativo quando os utilizadores não têm de apresentar resultados do seu trabalho. Por esse motivo, escrevemos dois livros, o primeiro com o título *Hints and Tips* («Sugestões e Apontamentos»), destinado ao

ZX80, e o segundo também com o título *Hints and Tips*, mas para o *ZX81*, os quais se destinavam a dar ao leitor algumas noções acerca da operação destes computadores.

Adoptámos, contudo, uma atitude diferente ao escrever este livro para o *ZX Spectrum*. Com 16 k de RAM como base, o utilizador sente menos necessidade de escrever os seus programas muito compactamente. Por outro lado, visto que dispõe de um conjunto de instruções e funções grandemente alargado, existe um incentivo muito maior para elaborar programas mais extensos e complicados.

Escrever bons programas é uma habilidade que tem de ser adquirida, e o nosso objectivo neste livro é ensiná-la através de exemplos. Os nossos programas ilustram uma série de técnicas bem conhecidas no mundo da informática — a utilização de registos de comprimento fixo e variável, a ordenação por comparações sucessivas (*bubble sort*, ou «ordenação de bolhas»), a pesquisa binária ou dicotómica (agenda e ficheiro de índices), etc. Outros programas exemplificam técnicas próprias do *Spectrum*, acentuando em particular a utilização eficiente da imagem. Na maioria, os programas estão divididos em várias secções, começando cada uma delas com um número de linha múltiplo de 500, de forma a que possamos acrescentar-lhe mais secções a fim de os adaptar às suas necessidades.

Esperamos que o leitor deste livro não tente lê-lo de um fôlego de uma ponta à outra. Em vez disso, deverá mergulhar nele em qualquer ponto que lhe atraia a atenção e experimentar o programa que aí esteja listado. Descrevemos os aspectos dos programas que nos parecem importantes, nuns casos com grande pormenor e, noutros, acerca dos quais se fazem observações semelhantes em outras partes do livro, com um tratamento menos completo.

Esperamos que o leitor tenha tanto gosto em utilizar os programas apresentados como a nós nos deu elaborá-los.

Andrew Hewson

CAPÍTULO 1

Programas úteis

FICHEIRO INDEXADO

O «Ficheiro indexado» é um programa de uso geral para armazenar informação. Este programa dá-nos a possibilidade de definir a natureza da informação a armazenar, de acrescentar ou apagar elementos, de apresentar a informação no *écran* ou na impressora, de aceder a determinado elemento de informação do ficheiro, de ordenar a informação por ordem alfabética ou numérica e de gravar *(save)* essa informação em *cassette* ou de a carregar em memória *(load)* a partir da *cassette*. O programa chama-se «Ficheiro indexado» por ter sido concebido de forma a executar as funções de um sistema de cartões perfurados, mas com muito maior velocidade, flexibilidade e comodidade.

Podemos controlar o programa introduzindo o código de um único carácter, a fim de seleccionar uma opção do *menu* impresso pela linha 200 e apresentado na figura 1. O código do carácter, armazenado em z$, é, em seguida, comparado sucessivamente, na linha 230, com cada um dos vários códigos. Se o código do carácter não coincidir com o código com o qual está a ser comparado, a igualdade correspondente, contida entre parênteses, toma o valor zero. Se coincidirem, o valor assumido pela igualdade é um. Um multiplicador à esquerda de cada par de parênteses dirige a execução do programa para a rotina adequada, a rotina em relação à qual ocorre uma coincidência de códigos. Por exemplo, se introduzirmos o carácter «P» para parar o programa, a última igualdade contida entre parênteses da linha 230 é substituída pelo valor um, pois z$="P" é, neste caso, uma proposição verdadeira. Todas as outras proposições entre parênteses tomam o valor zero, de modo a que a linha 230 seja executada como se contivesse a instrução GOTO 300+4200. O programa salta então para a linha 4500, imprime uma mensagem e pára.

```
Ficheiro Indexado

Menu

Definir estrutura de registo      D
Acrescentar registos              A
Eliminar registos                 E
Consultar registos                C
Imprimir registos                 I
Pesquisar registos                P
Ordenar registos                  O
Gravar (Save) registos            S
Carregar (Load) registos          L
Verificar                         V
Terminar                          T
```

Figura 1. *O* menu *do «Ficheiro indexado».*

Se introduzirmos um carácter que não figure no *menu,* não ocorre qualquer coincidência nas comparações de códigos e o programa salta para a linha 300, depois da qual é introduzida uma pausa e, terminada esta, o programa volta ao *menu.*

Cada uma das opções que figuram no dito *menu* é executada mais ou menos independentemente das outras e, portanto, o programa foi elaborado fragmentado em módulos distintos para cada opção. Para maior comodidade, o primeiro número de linha de cada módulo é um múltiplo de 500.

Definição de uma estrutura de registo Introduzir D

Suponhamos que o programa se destinava a arquivar os registos de um clube de utilizadores de microcomputadores. Os elementos a arquivar serão, nomeadamente, o apelido e uma inicial de cada sócio, a marca do computador que o sócio possui, a quota que ele deve pagar e se a tem ou não em dia. Como armazenaria esta informação o «Ficheiro indexado»?

O primeiro passo seria definir uma estrutura variável para conter a informação, seleccionando a opção «Definir estrutura de registos». O programa saltaria nesse caso para a linha 1000 e pediria que se introduzisse o número total de registos. É necessário um registo por sócio e, assim, por exemplo, se o clube tivesse 50 sócios, seriam necessários 50 registos. O

programa pediria então o número de campos alfabéticos (ou de cadeias) e o número de campos numéricos. Seria conveniente armazenar separadamente os nomes, iniciais, marcas de computadores e pagamento/não pagamento das quotas, constituindo na totalidade quatro campos alfabéticos. A quota de sócio constituiria o único campo numérico.

O programa pediria seguidamente o nome de cada um dos campos e, no caso dos campos alfabéticos, pedir-nos-ia que indicássemos o número máximo de caracteres a armazenar em cada um deles. O nome do campo admitirá um máximo de dez caracteres, mas não existe limite algum para o comprimento do campo.

Uma estrutura de registos adequada poderia apresentar-se como se vê na figura 2.

Número máximo de registos	=50
Número de campos alfabéticos	= 4
Número de campos numéricos	= 1

Número do campo alfabético	Nome	Comprimento
1	Sócio	20
2	Inicial	1
3	Computador	20
4	Pago?	1
Número do campo numérico	Nome	
1	Quota	

Figura 2. *Exemplo de uma estrutura de registos do «Ficheiro indexado». Esta estrutura é concebida de forma a conter o nome, inicial, marca do computador e mais pormenores acerca da condição de associado de cada um dos 50 membros de um clube de microinformática.*

Acrescentar registos Introduzir A

Esta opção introduz informação no computador logo após a estrutura de registo ter sido definida ou em qualquer ocasião

posterior. O programa imprime no *écran* o número do próximo registo livre e solicita a introdução, à vez, de cada um dos campos. Se tiver sido cometido um erro, a introdução de R faz o programa «voltar atrás», regressando ao campo anterior. Introduzindo T para terminar, obtém-se o retorno ao *menu*.

Eliminar registos Introduzir E

Esta opção apaga registos. O programa solicita os números do primeiro e do último registo a apagar. Assim, por exemplo, para apagar os três primeiros registos, deve introduzir-se 1 seguido de 3. Deve haver cuidado ao apagar mais do que um grupo de registos, pois o programa anula os intervalos deixados pelo apagamento, efectuando uma renumeração dos registos. A forma mais simples de evitar problemas consiste em apagar em primeiro lugar o grupo com o número mais elevado. Por exemplo, se se quiser apagar os registos 3 a 7 e 14 a 16, deve-se apagar primeiramente o último destes dois grupos.

Consultar e imprimir registos Introduzir C ou I

O programa pede os números do primeiro e do último registos a que se acede e apresenta seguidamente esses registos, e todos os que se situarem entre eles, no *écran* ou na impressora.

Pesquisar registos Introduzir P

O programa pede os números dos registos entre os quais deve efectuar a pesquisa e o nome do campo a pesquisar. Seguidamente, compara, um a um, a chave introduzida com o conteúdo do campo em cada registo e, se encontrar uma correspondência, imprime no *écran* o número do registo.

Ordenar registos («Bubble sort») Introduzir O

O programa solicita o número do campo que indica a ordem pela qual os registos devem ser ordenados. Se é um campo alfabético, ou de cadeia, a ordenação é efectuada alfabeticamente pelo código de carácter dos elementos do campo. Se é um campo numérico, os registos são ordenados por ordem numérica.

Gravar, carregar em memória e verificar Introduzir S, L ou V

Com estas opções grava-se a informação em *cassette*, carrega-se em memória *(load)* informação existente em *cassette* ou verifica-se uma cópia em *cassette*, comparando-a com a informação original existente em memória.

Mais informações acerca do programa

O «Ficheiro indexado» apresenta a maior parte das funções básicas exigidas a uma base de dados de computador, mas muitas mais poderíamos acrescentar. Por exemplo, seria útil, nomeadamente:

1) Editar (modificar, corrigir, actualizar) separadamente os campos contidos num registo;
2) Acrescentar novos campos a um conjunto de dados (informação) já existente;
3) Pesquisar simultaneamente vários campos;
4) Imprimir a informação em formatos especiais;
5) Aumentar o número de registos para além do limite inicialmente definido;
6) Criar novos conjuntos de dados (informação) a partir da selecção de determinados campos.

A primeira limitação levantada às funções de que se poderia dispor reside na área de RAM disponível. Na versão aqui apresentada, o programa ocupa cerca de 6,5 k dos 9 k disponíveis pelo utilizador no *Spectrum* de 16 k. Considerando que a informação em si requer uma quantidade de espaço considerável (50 registos de 20 caracteres cada um ocupam 1 k de memória, por exemplo), torna-se evidente a necessidade de se utilizar um computador de 48 k se for necessário acrescentar mais *software* ao programa principal. Uma alternativa consiste em segmentar o programa em blocos coerentes e acrescentar novos módulos a esses blocos. Cada um dos novos programas assim formados necessitaria de dispor da capacidade de gravar (SAVE), carregar em memória (LOAD) e verificar (VERIFY) a informação em *cassette* de modo a que cada programa pudesse «comunicar» com os outros, utilizando a *cassette* como meio de comunicação.

As variáveis mais importantes utilizadas pelo programa são as seguintes:

nn	—	número de campos numéricos
ns	—	número de campos de cadeias alfabéticas *(strings)*
cr	—	número corrente de registos
nr	—	número máximo de registos
f$(ns+nn,10)	—	quadro de cadeias *(string array)* que contém os nomes dos campos
lr	—	comprimento *(length)* total de todos os campos de cadeias alfabéticas
b$(nr,lr)	—	quadro de cadeias *(string array)* que contém toda a informação alfabética
b(ns,2)	—	quadro numérico que contém o endereço, em b$, do início e do fim de cada campo alfabético
a(nr,nn)	—	quadro numérico que contém toda a informação numérica
c(4)	—	quadro no qual são armazenados nn, ns, cr e nr quando passados para *cassette*

O último elemento de interesse é a ordenação por comparações sucessivas, dita «ordenação de bolhas» *(bubble sort)*. Este método de ordenação, à primeira vista pouco eficiente, é, na realidade, muito mais rápido do que outras técnicas bem mais complexas na aparência.

Numa ordenação deste tipo, o computador percorre os dados *(data)* verificando se cada um dos quadros *(arrays)*, quer numéricos (a(i,fn)), quer alfabéticos (b$(i,j)), que formam cada registo, é menor ou igual ao que o que se lhe segue. Caso contrário, trocam-se as posições relativas dos dois registos. O computador repete então o processo até que percorra os dados sem efectuar qualquer alteração, o que significa que a ordenação está concluída. Durante a ordenação, os registos menores «sobem» (como bolhas) uma posição de cada vez em direcção às posições cimeiras, e foi esta particularidade que deu origem ao nome desta técnica. Este método encontra-se muito difundido

devido a ser facilmente programável, rápido de executar e não requerer espaço de memória adicional para o seu funcionamento, para além da área ocupada por uma variável destinada a indicar em cada passagem se foi ou não efectuada alguma alteração.

Códigos de erro

Número de código	Significado
1	A letra de código introduzida não figura no menu.
2	Tentativa de definir uma estrutura de registo absurda, como, por exemplo, sem um único registo ou com um número de campos negativo.
3	Nome do campo com comprimento superior a 10 caracteres.
4	Comprimento de um campo alfabético inferior a 1.
5	Nome de ficheiro com comprimento superior a 10 caracteres.
6	Tentativa de pesquisa de campo inexistente.

Programa 1. *«Ficheiro indexado».*

```
  10 LET nn=0
  20 LET ns=0
  30 LET t=29
  40 LET cr=0
 200 CLS : PRINT TAB 8;"Ficheiro
 Indexado",,,TAB 14;"Menu",,,"De
finir estrutura de registo";TAB
t;"D","Acrescentar registos";TAB
 t;"A","Eliminar registos";TAB t
;"E","Consultar registos";TAB t;
"C","Imprimir registos";TAB t;"I
","Pesquisar registos";TAB t;"P"
,"Ordenar registos";TAB t;"O","G
ravar (Save) registos";TAB t;"S"
,"Carregar (Load) registos";TAB
t;"L","Verificar";TAB t;"V","Ter
minar";TAB t;"T"
 210 INPUT z#
 220 GO SUB 6000
```

```
 230 GO TO 300+5200*(z$="P")+470
0*(z$="E")+200*(z$="A")+700*(z$=
"D")+1200*(z$="C")+1700*(z$="I")
+2200*(z$="O")+2700*(z$="S")+320
0*(z$="L")+3700*(z$="U")+4200*(z
$="T")
 300 GO SUB 6050: PRINT "1"
 310 PRINT ,,"Pressione uma tecl
a p/ continuar"
 320 PAUSE 0
 330 GO TO 200
 510 PRINT ,,"Introduza, um a um
, os valores  para cada campo de
 cada registo",,,"Introduza :",,
"  F p/ terminar","  B p/ volta
r atras",,,
 520 FOR i=cr+1 TO nr
 530 PRINT "Registo num. ";i
 540 FOR j=1 TO ns+nn
 550 PRINT " Campo ";f$(j);
 560 INPUT z$;
 570 IF z$="F" OR z$="f" THEN PR
INT ,,,"Lancamentos terminados":
 GO TO 310
 580 IF j<>1 AND (z$="B" OR z$="
b") THEN LET j=j-1: GO TO 550
 590 IF j=1 AND z$="B" OR z$="b"
 THEN LET i=i-1: LET cr=cr-1: LE
T j=ns+nn: PRINT ,,,"Registo num
. ";i: GO TO 550
 600 IF j<=ns THEN LET b$(i,b(j,
1) TO b(j,2))=z$: GO TO 670
 610 LET a(i,j-ns)=VAL z$
 670 PRINT z$
 680 NEXT j
 690 LET cr=cr+1
 700 NEXT i
 710 PRINT ,,"Ficheiro repleto"
 720 GO TO 310
1000 PRINT ,,TAB 2;"Definir estr
utura de registo",,,"Introduza :
",,"  num. max de registos","  n
um. de campos alfanumericos","
num. de campos numericos",,,,
1010 INPUT nr;TAB 10;ns;TAB 20;n
n
1020 LET nr=INT nr: LET ns=INT n
s: LET nn=INT nn
1030 IF nr<=0 OR ns<0 OR nn<0 OR
 ns+nn=0 THEN GO SUB 6050: PRINT
 "2": GO TO 310
1040 DIM f$(ns+nn,10)
```



```
1050 LET lr=0
1060 IF ns=0 THEN GO TO 1250
1070 DIM b(ns,2)
1110 FOR i=1 TO ns
1120 PRINT "Campo alfanum.";i
1130 PRINT "Nome do campo ?  "
1140 INPUT z$: IF LEN z$>10 THEN
 GO SUB 6050: PRINT "3": GO TO 1
130
1150 LET f$(i)=z$
1160 PRINT AT 20,18;z$
1170 PRINT "Comprimento ?     "
1180 INPUT l: IF l<1 THEN GO SUB
 6050: PRINT "4": GO TO 1170
1190 PRINT AT 20,29;l: LET b(i,1
)=lr+1: LET lr=lr+l: LET b(i,2)=
lr
1200 NEXT i
1210 DIM b$(nr,lr)
1220 IF nn=0 THEN DIM a(1): GO T
O 1340
1250 FOR i=ns+1 TO ns+nn
1260 PRINT "Campo numerico ";i
1270 PRINT "Nome do campo ?"
1280 INPUT z$: IF LEN z$>10 THEN
 GO SUB 6050: PRINT "3": GO TO 1
270
1290 LET f$(i)=z$
1300 PRINT AT 20,18;z$
1310 NEXT i
1320 DIM a(nr,nn)
1330 DIM c(nn)
1340 IF nn<4 THEN DIM c(4)
1350 PRINT ,,"Definicao terminad
a"
1360 GO TO 310
1500 PRINT ,,TAB 7;"Consultar re
gistos"
1510 GO SUB 8500
1580 FOR i=b TO e
1590 PRINT "Registo ";i
1600 FOR j=1 TO ns
1610 PRINT "  ";f$(j);" : ";
1620 PRINT b$(i,b(j,1) TO b(j,2)
)
1630 NEXT j
1640 FOR j=ns+1 TO ns+nn
1650 PRINT "  ";f$(j);" : ";
1660 PRINT a(i,j-ns)
1670 NEXT j
1680 NEXT i
1710 PRINT ,,"Fim da consulta":
```

```
GO TO 310
2000 PRINT ,,TAB 7;"Imprimir reg
istos"
2010 GO SUB 8500
2080 FOR i=b TO e
2090 LPRINT "Registo ";i
2100 FOR j=1 TO ns
2110 LPRINT "  ";f$(j);" : ";
2120 LPRINT b$(i,b(j,1) TO b(j,2
))
2130 NEXT j
2140 FOR j=ns+1 TO ns+nn
2150 LPRINT "  ";f$(j);" : ";
2160 LPRINT a(i,j-ns)
2170 NEXT j
2180 NEXT i
2200 PRINT ,,"Impressao terminad
a": GO TO 310
2500 PRINT ,,TAB 8;"Ordenar regi
stos"
2505 IF cr=0 THEN PRINT "Ficheir
o vazio": GO TO 310
2510 PRINT ,,"Introduza o numero
  do campo que   deve comandar a
ordenacao :": INPUT fn
2520 LET fn=INT fn: IF fn<1 OR f
n>ns+nn THEN GO SUB 6050: PRINT
"6": GO TO 2510
2530 IF fn>ns THEN GO TO 2700
2540 LET in=0
2560 FOR i=1 TO cr-1
2570 FOR j=b(fn,1) TO b(fn,2)
2580 IF b$(i,j)<b$(i+1,j) THEN G
O TO 2660
2590 IF b$(i,j)<>b$(i+1,j) THEN
GO TO 2620
2600 NEXT j
2610 GO TO 2660
2620 LET z$=b$(i)
2630 LET b$(i)=b$(i+1)
2640 LET b$(i+1)=z$
2645 IF nn>0 THEN FOR j=1 TO nn:
 LET c(j)=a(i,j): LET a(i,j)=a(i
+1,j): LET a(i+1,j)=c(j): NEXT j
2650 LET in=in+1
2660 NEXT i
2670 IF in>0 THEN GO TO 2540
2680 PRINT ,,"Ordenacao terminad
a"
2690 GO TO 310
2700 LET in=0
2710 FOR i=1 TO cr-1
```



```
2720 IF a(i,fn-ns)<=a(i+1,fn-nsl
 THEN GO TO 2800
2730 FOR j=1 TO nn
2740 LET c(j)=a(i,j)
2750 LET a(i,j)=a(i+1,j)
2760 LET a(i+1,j)=c(j)
2770 NEXT j
2780 IF ns>0 THEN LET z$=b$(i):
LET b$(i)=b$(i+1): LET b$(i+1)=z
$
2790 LET in=in+1
2800 NEXT i
2810 IF in>0 THEN GO TO 2700
2820 PRINT ,,"Ordenacao terminad
a"
2830 GO TO 310
3000 PRINT ,,TAB 8;"Gravar fiche
iro",,,"Introduza o nome"
3010 INPUT z$: IF LEN z$>10 THEN
 GO SUB 6050: PRINT "5": GO TO 3
010
3020 LET c(1)=ns
3030 LET c(2)=nn
3040 LET c(3)=nr
3050 LET c(4)=cr
3060 PRINT AT 21,3; FLASH 1;"Gra
vacao a ser efectuada"
3070 SAVE z$ DATA a()
3080 SAVE z$ DATA b()
3090 SAVE z$ DATA b$()
3100 SAVE z$ DATA c()
3110 SAVE z$ DATA f$()
3120 PRINT AT 21,0;"Gravacao de
";z$,"terminada correctamente"
3130 GO TO 310
3500 PRINT ,,TAB 6;"Carregar um
ficheiro",,,"Introduza o nome"
3510 INPUT z$: IF LEN z$>10 THEN
 GO SUB 6050: PRINT "5": GO TO 3
510
3520 PRINT AT 21,2; FLASH 1;"Car
regamento a ser efectuado"
3530 LOAD z$ DATA a()
3540 LOAD z$ DATA b()
3550 LOAD z$ DATA b$()
3560 LOAD z$ DATA c()
3570 LOAD z$ DATA f$()
3580 PRINT AT 11,0;z$;" carregad
o em memoria",
3590 LET ns=c(1)
3600 LET nn=c(2)
3610 LET nr=c(3)
```

```
3620 LET cr=c(4)
3630 GO TO 310
4000 PRINT ,,TAB 5;"Verificar um
 ficheiro",,,"Introduza o nome"
4010 INPUT z$: IF LEN z$>10 THEN
 GO SUB 6050: PRINT "5": GO TO 4
010
4020 PRINT AT 21,3; FLASH 1;"Ver
ificacao a ser efectuada"
4030 VERIFY z$ DATA a()
4040 VERIFY z$ DATA b()
4050 VERIFY z$ DATA b$()
4060 VERIFY z$ DATA c()
4070 VERIFY z$ DATA f$()
4080 PRINT AT 11,0;z$;" correcta
mente verificado"
4090 GO TO 310
4500 PRINT ,,"O programa parou"
4510 STOP
5000 PRINT ,,TAB 8;"Eliminar reg
istos"
5010 GO SUB 8500
5070 FOR i=1 TO cr-e
5080 IF ns<>0 THEN LET b$(b+i-1)
=b$(e+i)
5090 IF nn<>0 THEN FOR j=1 TO nn
: LET a(b+i-1,j)=a(e+i,j): NEXT
j
5100 NEXT i
5110 LET cr=cr-e+b-1
5120 PRINT ,,"Eliminacao termina
da"
5130 GO TO 310
5500 PRINT ,,TAB 7;"Pesquisar re
gistos"
5510 GO SUB 8500
5520 PRINT ,,"Introduza o nome d
o campo a pes-quisar"
5530 INPUT z$: IF LEN z$>10 THEN
 LET z$=z$(1 TO 10)
5540 FOR i=1 TO ns+nn
5550 IF f$(i,1 TO LEN z$)=z$ THE
N GO TO 5600
5560 NEXT i
5570 GO SUB 8050: PRINT ;"6": GO
 TO 310
5600 PRINT ,,"Introduza o elemen
to a procurar"
5610 INPUT z$
5620 PRINT ,,z$;" aparece nos re
gistos num. :"
5630 IF i>ns THEN GO TO 5700
```

cls :-


```
5640 FOR j=b TO e
5650 FOR k=b(i,1) TO b(i,2)-LEN
z$+1
5660 IF z$=b$(j,k TO k+LEN z$-1)
 THEN PRINT TAB 6;j: GO TO 5680
5670 NEXT k
5680 NEXT j
5690 GO TO 310
5700 FOR j=b TO e
5710 IF a(j,i-ns)=VAL z$ THEN PR
INT TAB 6;j
5720 NEXT j
5730 GO TO 310
6000 LET z$=CHR$ (CODE z$-32*(CO
DE z$>96))
6010 RETURN
6050 BEEP .2,24
6060 PRINT "Erro ";
6070 RETURN
8500 IF cr=0 THEN PRINT ,,"Fiche
iro vazio": GO TO 310
8510 PRINT ,,"Introduza o numero
 do primeiro e  do ultimo regist
o a afectar"
8520 INPUT b,e
8530 LET b=1+(b-1)*(b>0)+(cr-b)*
(b>=cr)
8540 LET e=b+1+(e-b-1)*(e>=b)+(c
r-e)*(e>cr)
8550 RETURN
```

AGENDA

O programa anterior, o «Ficheiro indexado», reservava uma quantidade fixa de espaço para cada campo dentro do registo. O campo para o nome do sócio do clube de microcomputadores, por exemplo, ocupava 20 caracteres. Na maioria dos casos, alguns desses caracteres (ou melhor: desses espaços para caracteres) ficariam por utilizar, por o nome do sócio ter um comprimento inferior a 20 caracteres. Torna-se evidente que a técnica de armazenar informação em campos de comprimento fixo implica geralmente um desperdício de espaço.

Este programa («Agenda») foi elaborado com o fim de exemplificar a alternativa que consiste em permitir que o com-

primento varie de registo para registo. O apontamento relativo a um dia constitui um registo. As características mais importantes deste programa são as seguintes:

1) Os apontamentos podem ter qualquer comprimento.
2) Só se reserva espaço para dias em que haja apontamento a eles relativo.
3) Apontamentos relativos a um mesmo dia são automaticamente arquivados no mesmo registo.
4) Os apontamentos desactualizados são automaticamente eliminados.
5) Usa-se uma técnica de pesquisa binária para localizar o apontamento relativo a qualquer data que tenha sido introduzida.

Estrutura do programa

Na tabela 1 especifica-se a função de cada uma das variáveis principais.

Variável	Conteúdo
tr	Número total de apontamentos permitido.
z$	Quadro de cadeias utilizado para conter os apontamentos.
tz	Comprimento máximo de z$.
p(2,tr+1)	Quadro numérico. O primeiro elemento de cada par armazena a data num código numérico. O segundo elemento indica a localização do apontamento em z$.
nr	Número actual ou corrente de apontamentos.
m$	Quadro de cadeias que contêm o número de dias de cada mês.

Tabela 1. *Nome e função de cada uma das variáveis principais.*

O programa utiliza repetidamente duas sub-rotinas: uma destina-se a codificar a data, e a outra procura a informação que constitui o apontamento relativo a uma dada data. A rotina da «data» é constituída pelas linhas 400 a 630, e pede-nos que introduzamos a data sob a forma seguinte:

Dia do mês em dois dígitos
Nome do mês em letras — só as primeiras três são significativas
Ano, com ou sem os dois primeiros dígitos (19 — —)

Eis várias formas válidas de data:

19 Dezembro 1985
19 DEZ 1985
19 dezem 1985

A rotina mencionada verifica a sintaxe da data, incrementa o segundo elemento de «d» se o ano em causa for um ano bissexto e armazena uma forma numérica da data na variável «da» utilizando um código simples. Assim, por exemplo, 19 Dezembro 1982 é armazenado na forma de 21219.

A rotina de «pesquisa» situa-se nas linhas 7000 a 7270. Todos os lançamentos em z$ são mantidos pela ordem das datas respectivas, e a forma numérica da data correspondente a cada apontamento é armazenada nos primeiros elementos do quadro bidimensional «p». Deste modo, para determinar se existe algum apontamento relativo a uma dada data armazenada em «da», é necessário comparar «da» com cada um dos primeiros elementos em «p». A rotina seguinte exemplifica um método simples de pesquisar «p» («nr» representa o número de apontamentos na agenda):

```
1Ø  FOR i=1 TO nr
2Ø  IF da=p(1,i) THEN PRINT «ENCONTRADO UM APONTAMENTO»: STOP
3Ø  NEXT i
4Ø  PRINT «NÃO HA' APONTAMENTOS»
5Ø  STOP
```

Esta técnica revela-se lenta por ser necessário verificar cada valor de p(1,i). A rotina de «pesquisa» utiliza um método mais rápido, a «pesquisa binária», baseada em que os apontamentos estão ordenados por datas. Eis o método:

1) Determina-se o elemento central da zona de «p» a pesquisar;
2) Compara-se «da» com o elemento central de «p».
3) Se «da» for menor do que o elemento central, o programa «debruça-se» sobre a metade inferior de «p» e volta a 1).
4) Se «da» for maior do que o elemento central, o programa «debruça-se» sobre a metade superior de «p» e volta a 1).

Suponhamos, por exemplo, que «da» é igual a p(1,47) e que o número de apontamentos é 83. A pesquisa far-se-ia assim:

1) Compara-se «da» com p(1,83/2), isto é,«da» com p(1,41).
2) «da» é maior do que o elemento p(1,41) e, portanto, compara-se seguidamente «da» com $p(1,41 + \frac{83-41}{2}) = p(1,62)$:
3) «da» é menor do que o elemento p(1,62) e, portanto, compara-se seguidamente «da» com $p(1,41 + \frac{62-41}{2}) = p(1,51)$.
4) «da» é menor do que o elemento p(1,51) e, portanto, compara-se seguidamente «da» com $p(1,41+\frac{51-41}{2})= p(1,46)$.
5) «da» é maior do que o elemento p(1,46) e, portanto, compara-se seguidamente «da» com $p(1,46+\frac{51-46}{2})= p(1,48)$.
6) «da» é menor do que o elemento p(1,48) e, portanto, compara-se seguidamente «da» com $p(1,46+\frac{48-46}{2})= p(1,47)$.
7) «da» = p(1,47).

Pode verificar-se que se concluiu a pesquisa em sete passos, em vez dos 47 requeridos pela técnica anterior. Em cada passo da pesquisa, o conjunto de valores possíveis é reduzido a metade e, deste modo, o número máximo de passos necessários para localizar um apontamento é dado pela equação

$$nr<2^m$$

onde nr é o número de apontamentos a pesquisar e m é o menor número inteiro para o qual a desigualdade é verdadeira.

Como utilizar a «Agenda»

Quando é executado pela primeira vez, o programa define os quadros *(arrays)* de que necessita e coloca os nomes e o número de dias de cada mês, obtidos na declaração DATA da linha 200, em m$ e d, respectivamente. Tendo inicializado o programa desta forma, não devemos fazer executá-lo subsequentemente por meio de RUN, pois, se o fizermos, as variáveis e o seu conteúdo perder-se-ão. Para que tal não aconteça, devemos introduzir GOTO 3000 em vez de RUN.

O programa solicita a data actual por meio da sub-rotina da «data» e compara-a com as datas dos apontamentos no quadro «indicador» por meio da rotina de «pesquisa». Se existirem apontamentos com datas anteriores à data actual ou corrente, esses apontamentos são eliminados (linhas 900 a 970).

A linha 1010 imprime um *menu* como o da figura 3 e convida-nos a seleccionar uma opção. As opções de «consulta», «impressão» e «eliminação» de apontamentos têm um funcionamento bastante simples: as rotinas da «data» e da «pesquisa» localizam o elemento pretendido e apresentam-no no *écran* ou na impressora, ou apagam-no, conforme for necessário.

```
        Agenda Spectrum

      11 JUNHO     1985

            Menu

Acrescentar um apontamento      A
Consultar um apontamento        C
Imprimir um apontamento         I
Eliminar um apontamento         E
Parar o programa                P
```

Figura 3. *O* menu *impresso por «Agenda»*

Na figura 3 apresenta-se um exemplo de *output* do programa. Os apontamentos relacionam-se com uma visita à República Federal da Alemanha.

A opção «acrescentar um apontamento» é um pouco mais complexa e utiliza também as sub-rotinas da «data» e da «pesquisa». Se já existir um apontamento na data em questão, o novo apontamento é acrescentado ao seu final, com uma vírgula e um espaço em branco entre eles como separadores. Mas se não existir qualquer apontamento na data pretendida, o programa introduz o novo apontamento sob essa data e na ordem a ela relativa.

Tal como é apresentado, o programa reserva espaço para 90 apontamentos, totalizando 1800 caracteres, e utiliza a quase totalidade do espaço disponível num *Spectrum* de 16 k. Contudo, com o *Spectrum* de 48 k, os valores de tr e tz (linhas 100 e 110) podem ser aumentados para 1100 e 22000, respectivamente.

```
Apontamento de 11 JUNHO      1985

14:30  Partida para o aeroporto
da Portela. 16:15 Voo Lufthansa
LH 201 para Frankfurt.

Apontamento de 12 JUNHO      1985

Visita a Karlsruhe

Apontamento de 14 JUNHO      1985

Visita a Munique.

Apontamento de 18 JUNHO      1985

Visita a Wiesbaden.

Apontamento de 20 JUNHO      1985

Visita a Saarbrucken.

Apontamento de 21 JUNHO      1985

13:10 Voo Lufthansa LH 200 para
Lisboa.
```

Figura 4. *Itinerário de uma visita à Alemanha Ocidental impresso pelo programa «Agenda»*

Programa 2. *«Agenda»*

```
  50 REM "AGENDA"
 100 LET tr=90
 110 LET tz=1800
 120 DIM p(2,tr+1)
 130 DIM z$(1,tz)
 140 LET nr=0
 200 DATA "JANEIRO","FEVEREIRO",
"MARCO","ABRIL","MAIO","JUNHO","
JULHO","AGOSTO","SETEMBRO","OUTU
BRO","NOVEMBRO","DEZEMBRO",31,28
,31,30,31,30,31,31,30,31,30,31
 210 DIM m$(12,9): DIM d(12)
 220 FOR i=1 TO 12
 230 READ m$(i)
 240 NEXT i
 250 FOR i=1 TO 12
 260 READ d(i)
 270 NEXT i
 300 PRINT TAB 9;"Agenda Spectru
m",,,"Introduza a data de hoje:"
 310 GO SUB 400
 320 LET d$=STR$ dia+" "+m$(i)+"
 "+STR$ y
 330 GO SUB 7000
 340 IF nr=0 THEN GO TO 1000
 350 IF e<0 THEN LET e=-e-1: GO
TO 900
 360 LET e=e\
 370 GO TO 900
 410 PRINT ,,"Introduza o dia, o
 mes e o ano :"
 420 PRINT ,,"Dia = ",
 430 INPUT dia
 440 PRINT dia
 450 PRINT "Mes = ",
 460 INPUT n$: IF LEN n$<3 THEN
BEEP 1,12: GO TO 460
 470 FOR i=1 TO 3
 480 LET n$(i)=CHR$ (CODE n$(i)-
32*(n$(i)>"f"))
 490 NEXT i
```

```
 500 FOR i=1 TO 12
 510 IF n$(1 TO 3)=m$(i,1 TO 3)
THEN GO TO 540
 520 NEXT i
 530 BEEP 1,12: GO TO 460
 540 PRINT m$(i)
 550 PRINT "Ano = ",
 560 INPUT y: IF y<80 OR y>2000
THEN BEEP 1,12: GO TO 560
 570 IF y>100 AND y<1980 THEN BE
EP 1,12: GO TO 560
 580 LET y=y+1900*(y<1980)
 590 PRINT y
 600 LET d(2)=28+(y=4*INT (y/4)
AND y<>2000)
 610 IF dia<1 OR dia>d(i) THEN B
EEP 1,12: PRINT ,,"""Dia"" incor
recto - tente de novo": PAUSE 10
0: GO TO 400
 620 LET da=dia+i*100+(y-1980)*1
0000
 630 RETURN
 900 LET len=p(2,e+1)
 910 LET z$(1,1 TO p(2,nr+1)-len
)=z$(1,p(2,e+1)+1 TO p(2,nr+1))
 920 LET nr=nr-e
 930 FOR i=1 TO nr
 940 LET p(1,i)=p(1,e+i)
 950 LET p(2,i)=p(2,e+i)-len
 960 NEXT i
 970 LET p(2,nr+1)=p(2,nr+e+1)-l
en
1000 CLS
1010 PRINT TAB 9;"Agenda Spectru
m"'';TAB 8;d$,,TAB 14;"Menu",,,"
Acrescentar um apontamento";TAB
31;"A";"Consultar um apontamento
";TAB 31;"C";"Imprimir um aponta
mento";TAB 31;"I";"Eliminar um a
pontamento";TAB 31;"E";"Parar o
programa";TAB 31;"P"
1020 INPUT a$: IF CODE a$>96 THE
N LET a$=CHR$ (CODE a$-32)
1030 GO TO 1100+400*(a$="A")+900
*(a$="C")+1400*(a$="I")+1900*(a$
="E")+2400*(a$="P")
1100 PRINT ,,"Pres. uma tecla p/
voltar ao menu"
1110 PAUSE 0
1120 GO TO 1000
1500 PRINT ,,TAB 3;"Acrescentar
um apontamento",,,"Introduza a d
```



```
ata do apontamento"
1510 GO SUB 400
1520 LET e$=STR$ dia+" "+m$(i)+"
 "+STR$ y
1530 PRINT ,,"Escreva o apontame
nto:"
1540 INPUT w$
1550 LET len=LEN w$
1600 IF nr<>0 THEN GO TO 1700
1610 LET nr=nr+1
1620 LET p(1,1)=da
1630 LET p(2,1)=0: LET p(2,2)=le
n
1640 LET z$(1,1 TO len)=w$
1650 GO TO 1100
1700 GO SUB 7000
1710 IF e<0 THEN LET e=-e: GO TO
 1850
1720 LET e=eu
1750 LET nr=nr+1
1760 LET p(2,nr+1)=p(2,nr)+len
1770 FOR i=nr TO e+1 STEP -1
1780 LET p(1,i)=p(1,i-1): LET p(
2,i)=p(2,i-1)+len
1790 NEXT i
1800 LET p(1,e)=da
1810 LET z$(1,1 TO p(2,nr+1))=z$
(1,1 TO p(2,e))+w$+z$(1,p(2,e)+1
 TO p(2,nr+1))
1820 GO TO 1100
1850 LET z$(1,1 TO p(2,nr+1)+len
+2)=z$(1,1 TO p(2,e+1))+": "+w$+
z$(1,p(2,e+1)+1 TO p(2,nr+1))
1860 LET p(2,nr+1)=p(2,nr+1)+len
+2
1870 FOR i=nr TO e+1 STEP -1
1880 LET p(2,i)=p(2,i)+len+2
1890 NEXT i
1900 GO TO 1100
2000 PRINT ,,TAB 4;"Consultar um
 apontamento",,,"Introduza a dat
a do apontamento"
2010 GO SUB 400: GO SUB 7000
2020 LET e$=STR$ dia+" "+m$(i)+"
 "+STR$ y
2030 IF e<0 THEN LET e=-e: GO TO
 2050
2040 PRINT ,,"Nao ha' ap. em ";e
$: GO TO 1100
2050 PRINT ,,"Apontamento de ";e
$
2060 PRINT ,,z$(1,p(2,e)+1 TO p(
```

```
2,e+1))
2070 GO TO 1100
2500 PRINT ,,TAB 4;"Imprimir um
apontamento",,,"Introduza a data
 do apontamento"
2510 GO SUB 400: GO SUB 7000
2520 LET e$=STR$ dia+" "+m$(i)+"
 "+STR$ y
2530 IF e<0 THEN LET e=-e: GO TO
 2550
2540 LPRINT ,,"Nao ha' ap. em ";
e$: GO TO 1100
2550 LPRINT ,,"Apontamento de ";
e$
2560 LPRINT ,,z$(1,p(2,e)+1 TO p
(2,e+1))
2570 GO TO 1100
3000 PRINT ,,TAB 4;"Eliminar um
apontamento",,,"Introduza a data
 do apontamento"
3010 GO SUB 400: GO SUB 7000
3020 LET e$=STR$ dia+" "+m$(i)+"
 "+STR$ y
3030 IF e<0 THEN LET e=-e: GO TO
 3050
3040 PRINT ,,"Nao ha' ap. em ";e
$: GO TO 1100
3050 LET len=p(2,e+1)-p(2,e)
3060 LET z$(1,1 TO p(2,nr+1)-len
)=z$(1,1 TO p(2,e))+z$(1,p(2,e+1
)+1 TO p(2,nr+1))
3070 LET nr=nr-1
3080 FOR i=e TO nr
3090 LET p(1,i)=p(1,i+1)
3100 LET p(2,i)=p(2,i+1)-len
3110 NEXT i
3120 LET p(2,nr+1)=p(2,nr+2)-len
3130 PRINT "Registo eliminado":
GO TO 1100
3500 PRINT ,,"O programa parou"
3510 STOP
6000 PRINT "da",: INPUT da: PRIN
T da
6040 LET nr=10
6100 FOR i=1 TO 10
6110 PRINT i,p(1,i)
6120 NEXT i
7000 LET el=0: LET eu=nr+1
7010 IF eu=el+1 THEN RETURN
7020 LET e=el+INT ((eu-el)/2)
7030 GO TO 7050+100*(da<p(1,e))+
200*(da>p(1,e))
```



```
7050 LET e=-e: RETURN
7060 STOP
7160 LET eu=e
7170 GO TO 7010
7260 LET el=e
7270 GO TO 7010
```

CALENDÁRIO

Este programa gera o calendário de qualquer mês ou ano entre 1980 e o ano 2000. O calendário de cada mês é apresentado no *écran* ou na impressora como uma grelha de 7 × 5 quadrados, com cada dia impresso no canto inferior esquerdo de cada quadrado e o mês, o ano e os dias da semana impressos no topo da grelha.

Este programa foi escrito pouco depois do da «Agenda» e, por isso, tem com ele em comum várias rotinas e variáveis. Assim, por exemplo, o nome e o número de dias de cada mês são colocados em m$ e d, respectivamente, e obtidos de uma declaração DATA. Também coloca os dias da semana em d$ a partir da mesma declaração DATA.

Ao executarmos o programa, temos a possibilidade de o *output* aparecer apenas no *écran* ou também na impressora. Seleccionada a opção, o programa pede-nos a introdução do mês (se o calendário for apenas de um mês) e do ano cujo calendário desejamos. Só as três primeiras letras do nome do mês são significativas, e podemos introduzi-las como maiúsculas ou como minúsculas. Do mesmo modo, podemos omitir os dois primeiros dígitos do ano, se o desejarmos. Desta forma, qualquer das versões seguintes é aceitável:

Setem 85
SETEM 85
set 1985

A figura 5 apresenta o exemplo de um calendário para Dezembro de 1985, tal como este programa o imprime. A disposição usual consiste em iniciar a grelha do calendário de

cada mês na segunda-feira da semana em que o mês começa. No caso da figura, o primeiro dia do mês é uma quarta-feira e, sendo o dito mês de 31 dias, aparecem dois quadrados vazios na grelha logo antes do início e logo após o fim do mês.

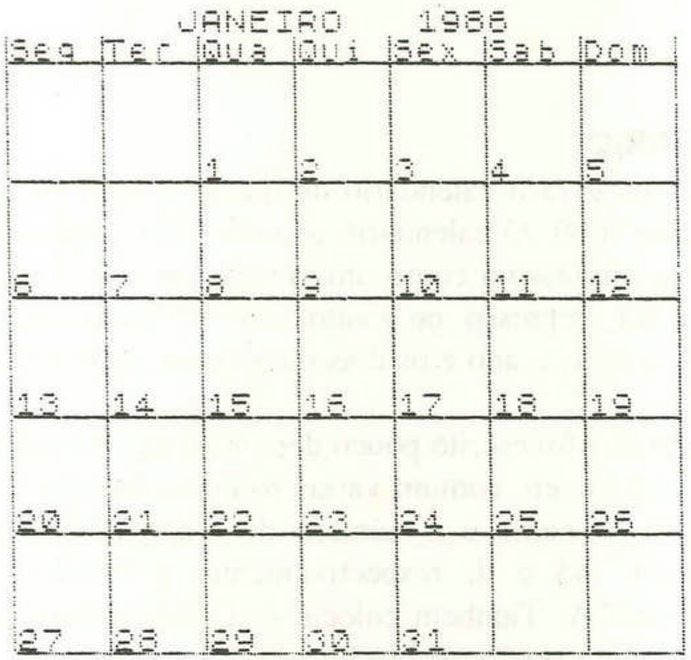


Figura 5. *Calendário para Janeiro de 1986.*

Pode, contudo, acontecer que o primeiro dia de um mês de 31 dias calhe a um sábado ou a um domingo. Se a grelha do calendário desse mês fosse começar na segunda-feira anterior a esse sábado ou domingo, haveria cinco ou seis quadrados vazios antes do início do mês e o trigésimo ou o trigésimo primeiro dia do mês (ou ambos) necessitariam de um trigésimo sexto e de um trigésimo sétimo quadrados, respectivamente.

Para resolver esta situação, colocam-se nove dias, e não sete, no quadro de cadeias dos dias da semana, d$, sendo os dois primeiros e os dois últimos «Sab» e «Dom». Se um mês tiver início num sábado ou num domingo, o programa reserva o primeiro quadrado da grelha para esse dia. O resultado é exemplificado pela figura 6, onde se apresenta o calendário para Junho de 1985.

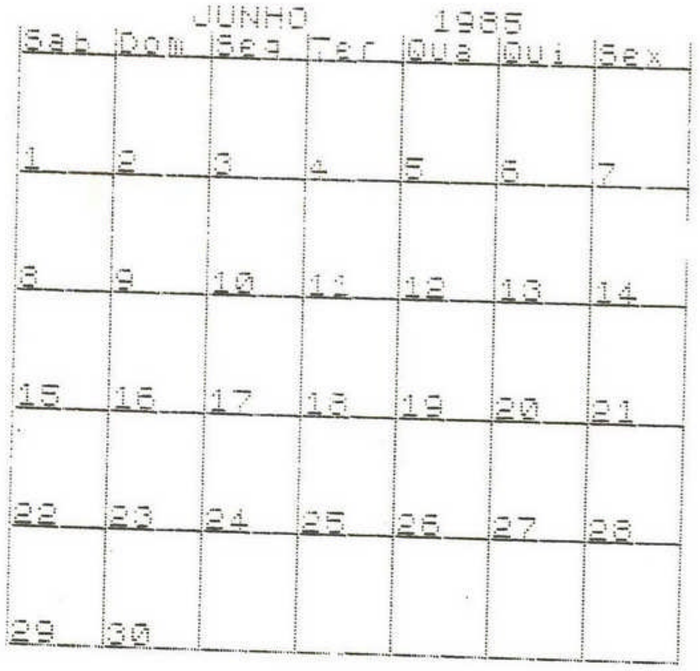


Figura 6. *Calendário para Junho de 1985.*

Programa 3. *«Calendário»*

```
  5 REM "CALENDARIO"
 10 PRINT TAB 6;"Calendario Spe
ctrum"
 20 PRINT ,,"Deseja o calendari
o de um mes   ou de um ano (M ou
 A) ?"
 30 INPUT z$: LET z$=CHR$ (CODE
 z$-32*(z$>"ï")): IF z$<>"M" AND
 z$<>"A" THEN BEEP 1,12: GO TO 3
0
 40 PRINT ,,"Deseja uma copia n
a impressora  (S ou N) ?"
 50 INPUT c$: LET c$=CHR$ (CODE
 c$-32*(c$>"ï")): IF c$<>"S" AND
 c$<>"N" THEN BEEP 1,12: GO TO 5
0
 100 DATA "JANEIRO","FEVEREIRO",
"MARCO","ABRIL","MAIO","JUNHO","
JULHO","AGOSTO","SETEMBRO","OUTU
BRO","NOVEMBRO","DEZEMBRO","Sab"
,"Dom","Seg","Ter","Qua","Qui","
Sex","Sab","Dom",31,28,31,30,31,
```

```
30,31,31,30,31,30,31
  110 DIM m$(12,9): DIM d$(9,3):
DIM d(12)
  120 FOR i=1 TO 12
  130 READ m$(i)
  140 NEXT i
  150 FOR i=1 TO 9
  160 READ d$(i)
  170 NEXT i
  180 FOR i=1 TO 12
  190 READ d(i)
  200 NEXT i
  300 CLS
  310 IF z$="A" THEN GO TO 3000
  500 PRINT TAB 6;"Calendario de
um mes",,,"Introduza mes e ano"
  510 PRINT ,,"Mes = ",
  520 INPUT n$: IF LEN n$<3 THEN
BEEP 1,12: GO TO 520
  530 FOR i=1 TO 3
  540 LET n$(i)=CHR$ (CODE n$(i)-
32*(n$(i)>"f"))
  550 NEXT i
  560 FOR i=1 TO 12
  570 IF n$(1 TO 3)=m$(i,1 TO 3)
THEN GO TO 600
  580 NEXT i
  590 BEEP 1,12: GO TO 510
  600 PRINT m$(i)
  610 PRINT "Ano = ",
  620 INPUT y: IF y<80 OR y>2000
THEN BEEP 1,12: GO TO 620
  630 IF y>100 AND y<1980 THEN BE
EP 1,12: GO TO 620
  640 LET y=y+1900*(y<1980)
  650 PRINT y
  660 PAUSE 50
  670 GO SUB 700
  680 STOP
  700 LET j=0
  710 IF i=1 THEN GO TO 800
  720 LET d(2)=28+(y=4*INT (y/4)
AND y<>2000)
  730 FOR l=1 TO i-1
  740 LET j=j+d(l)
  750 NEXT l
  800 LET j=j+4+(y-1980)*365+INT
((y-1981)/4)
  810 LET j=j+1-7*INT (j/7)
 1000 CLS
 1010 PRINT TAB 9;m$(i);TAB 19;y
 1100 LET b=-2+4*(j-2+(3-j)*(j<3)
)
```



```
 1110 LET a=5
 1120 FOR l=1 TO d(i)
 1130 PRINT AT a,b;l
 1140 LET b=b+4
 1150 IF b>26 THEN LET b=2: LET a
=a+4
 1160 NEXT l
 1500 LET j=j-(j-3)*(j>3)
 1510 LET a=2
 1520 FOR l=j TO j+6
 1530 PRINT AT 1,a;d$(l)
 1540 LET a=a+4
 1550 NEXT l
 2000 FOR a=15 TO 239 STEP 32
 2010 PLOT a,0
 2020 DRAW 0,167
 2030 NEXT a
 2050 FOR a=0 TO 167 STEP 32
 2060 PLOT 15,a
 2070 DRAW 224,0
 2080 NEXT a
 2090 IF c$="s" THEN COPY
 2100 RETURN
 3000 PRINT TAB 6;"Calendario de
um ano",,,"Introduza o ano"
 3010 PRINT ,,"Ano = ",
 3020 INPUT y: IF y<80 OR y>2000
THEN BEEP 1,12: GO TO 3020
 3030 IF y>100 AND y<1980 THEN BE
EP 1,12: GO TO 3020
 3040 LET y=y+1900*(y<1980)
 3050 PRINT y
 3060 PAUSE 50
 3100 IF c$="s" THEN LPRINT TAB 6
;"Calendario para ";y: LPRINT :
LPRINT
 3110 FOR i=1 TO 12
 3120 GO SUB 700
 3130 IF c$="s" THEN LPRINT : LPR
INT : GO TO 3150
 3140 PRINT #0;"Pressione uma tec
la p/ continuar": PAUSE 0
 3150 NEXT i
```

CAPÍTULO 2

Programas recreativos

ENFORCADO

Esta versão do conhecido jogo de palavras começa com um vocabulário muito limitado, mas tem a capacidade de aprender depressa. Para ampliar o vocabulário do programa, «ensinando-lhe» novas palavras, poderemos escolher entre introduzir essas palavras explicitamente ou, implicitamente, jogando o jogo. No segundo caso, o programa acrescenta ao seu vocabulário as palavras que ainda desconhecia.

As linhas 10 a 95 definem vários parâmetros que determinam as dimensões e a localização do «enforcado» no *écran*. A figura é traçada chamando cada uma das rotinas situadas a partir da linha 8000, à vez, de acordo com o valor da variável sub.

O programa coloca o seu vocabulário inicial, obtido da declaração DATA, no quadro de cadeias z$. O número e o comprimento das palavras na declaração DATA não têm qualquer importância relativamente ao tamanho de z$, existindo apenas a condição de elas terem de ser escritas em maiúsculas e de a última palavra ter de ser «FIM».

O *menu* é impresso pela linha 1000 e apresentado na figura 7.

```
                ENFORCADO

Aumentar vocabulario                    A
O Spectrum adivinha 1 palavra           S
Voce adivinha 1 palavra                 V
Gravar (SAVE) vocabolario               G
Carregar (LOAD) vocabulario             C
Parar programa                          P
```

Figura 7. *O* menu *do «Enforcado».*

Programa 4. *«Enforcado»*

```
  5 REM "ENFORCADO"
 10 LET p1=0: LET p2=24
 15 LET p3=32: LET p4=16
 20 LET p5=p1: LET p6=p2+p4
 25 LET p7=128
 30 LET p8=p6+p7
 35 LET p9=72
 40 LET p10=p5+p9
 45 LET p11=24
 50 LET p12=8: LET p13=p8-p11-p
12
 55 LET p14=p13-p12: LET p15=32
 60 LET p16=p13-2*p12
 65 LET p17=24: LET p18=8
 70 LET p19=p14-p15
 75 LET p20=24: LET p21=40
200 LET lz=3000
210 DIM z$(lz+2)
220 LET ez=0
230 LET nz=0
240 DATA "BYTE","MEMORIA","SPEC
TRUM","PROGRAMA","COMPUTADOR","H
EXADECIMAL","FIM"
250 READ x$
260 LET z$( TO ez+LEN x$+1)=z$(
 TO ez)+CHR$ LEN x$+x$
270 LET ez=ez+LEN x$+1
280 LET nz=nz+1
290 IF x$<>"FIM" THEN GO TO 250
1000 CLS : PRINT ,,TAB 11;"ENFOR
CADO"
1010 PRINT ,,,,"Aumentar vocabul
ario";TAB 31;"AO Spectrum adivin
ha 1 palavra";TAB 31;"SVoce adiv
inha 1 palavra";TAB 31;"VGravar
(SAVE) vocabolario";TAB 31;"GCar
```



```
regar (LOAD) vocabulario";TAB 31
;"CParar programa";TAB 31;"P"
1020 INPUT y$: LET y$=CHR$ (CODE
 y$-32*(y$>"£"))
1030 GO TO 1100+400*(y$="A")+900
*(y$="S")+1900*(y$="V")+1400*(y$
="G")+2400*(y$="C")+2900*(y$="P"
)
1100 INPUT "Pressione ENTER p/ c
ontinuar ";y$
1110 GO TO 1000
1500 PRINT ,,TAB 6;"Aumentar voc
abulario"
1510 PRINT ,,"O vocabulario actu
al e':"
1520 PRINT
1530 LET pw=1
1540 PRINT z$(pw+1 TO CODE z$(pw
)+pw);", ";
1550 LET pw=pw+1+CODE z$(pw)
1560 IF pw<ez THEN GO TO 1540
1570 PRINT
1600 INPUT "Introduza uma palavr
a ou prima  ENTER p/ terminar ";
x$
1610 IF x$="" THEN GO TO 1000
1620 IF ez+LEN x$>=lz THEN PRINT
 : PRINT ,,"Nao sobra mais espac
o": GO TO 1100
1630 FOR i=1 TO LEN x$
1640 LET x$(i)=CHR$ (CODE x$(i)-
32*(x$(i)>"£"))
1650 NEXT i
1660 LET z$( TO ez+1+LEN x$)=z$(
 TO ez)+CHR$ LEN x$+x$
1670 LET ez=ez+1+LEN x$
1680 LET nz=nz+1
1690 GO TO 1600
2000 CLS : PRINT TAB 3;"O Spectr
um adivinha 1 palavra"
2010 INPUT "Quantas letras ? ";n
l
2020 IF nl<0 THEN BEEP .2,24: GO
 TO 2010
2030 PRINT AT 5,16;nl;" letras"
2040 PRINT AT 20,(32-nl)/2;
2050 FOR i=1 TO nl
2060 PRINT "-";
2070 NEXT i
2080 PRINT
2090 LET s=0
2100 LET sub=7900
```

```
2110 DIM m(26)
2120 LET n=1+INT (26*RND)
2130 IF m(n)=1 THEN GO TO 2120
2140 LET m(n)=1
2150 LET x$=CHR$ (64+n)
2170 PRINT AT 21,4;"A palavra co
ntem ";x$;" ?"
2180 INPUT y$: IF LEN y$<1 THEN
BEEP .2,24: GO TO 2180
2190 IF y$(1)="S" OR y$(1)="s" T
HEN GO TO 2400
2200 LET sub=sub+100
2210 GO SUB sub
2220 IF sub<8900 THEN GO TO 2120
2230 PRINT AT 7,15; FLASH 1;"BEM
 FEITO!"
2240 PAUSE 300
2250 INPUT "Qual e'a palavra ? "
;x$
2260 IF LEN x$<>nl THEN BEEP .2,
24: GO TO 2250
2270 LET pw=1
2275 FOR i=1 TO nl: LET x$(i)=CH
R$ (CODE x$(i)-32*(x$(i)>"£")):
NEXT i
2280 IF x$=z$(pw+1 TO pw+CODE z$
(pw)) THEN GO TO 1100
2290 LET pw=pw+CODE z$(pw)+1
2300 IF pw<ez THEN GO TO 2280
2310 IF ez+nl+1>=lz THEN GO TO 1
100
2320 LET z$( TO ez+nl+1)=z$( TO
ez)+CHR$ nl+x$
2330 LET ez=ez+nl+1
2340 LET nz=nz+1: GO TO 1100
2400 INPUT "Quantas vezes ela oc
orre? ";n
2410 IF n<0 OR n>nl THEN BEEP .2
,24: GO TO 2400
2420 LET m=0
2430 INPUT "Qual a posicao da le
tra? ";i: IF i<1 OR i>nl THEN BE
EP .2,24: GO TO 2430
2440 PRINT AT 20,(32-nl)/2+i-1;x
$
2450 LET m=m+1: IF m<n THEN GO T
O 2430
2460 LET s=s+1: IF s<nl THEN GO
TO 2120
2470 PRINT AT 7,16; FLASH 1;"Pou
ca sorte": PAUSE 300: GO TO 1100
2500 PRINT ,,TAB 6;"Gravar vocab
```



```
ulario"
2510 INPUT "Introduza o nome ";y
$
2520 IF LEN y$>10 THEN BEEP .1,2
4: GO TO 2510
2530 LET z$(lz+1)=CHR$ (ez-256*I
NT (ez/256))
2540 LET z$(lz+2)=CHR$ (INT (ez/
256))
2550 SAVE y$ DATA z$()
2560 INPUT "Rebobine a fita - pr
ima ENTER";x$
2570 VERIFY y$ DATA z$()
2580 GO TO 1100
3000 LET n=1+INT (nz*RND)
3010 CLS : PRINT ,,TAB 12;"ENFOR
CADO"
3020 LET nw=1: LET pw=1
3030 IF nw=n THEN GO TO 3100
3040 LET nw=nw+1: LET pw=pw+CODE
 z$(pw)+1
3050 GO TO 3030
3100 LET nl=CODE z$(pw)
3110 PRINT AT 5,16;nl;" letras"
3120 PRINT AT 20,(32-nl)/2;
3130 FOR i=1 TO nl
3140 PRINT "_";
3150 NEXT i
3160 PRINT
3170 LET x$=z$(pw+1 TO pw+CODE z
$(pw))
3180 DIM l(LEN x$)
3190 LET sub=7900
3200 INPUT "Introduza uma letra
";y$
3205 LET y$=CHR$ (CODE y$-32*(y$
 £"))
3210 LET in=0: LET s=0
3220 FOR i=1 TO LEN x$
3230 IF l(i)=0 AND x$(i)=y$(1) T
HEN LET l(i)=1: PRINT AT 20,(32-
nl)/2+i-1;y$: LET in=1
3240 LET s=s+l(i)
3250 NEXT i
3260 IF in=0 THEN LET sub=sub+10
0: GO SUB sub
3270 IF s=LEN x$ THEN PRINT AT 7
,16; FLASH 1;"CORRECTO": PRINT F
LASH 1;AT 9,16;89-sub/100;" pont
os": PRINT AT 20,(32-nl)/2; FLAS
H 1;x$: GO TO 1100
3280 IF sub<8900 THEN GO TO 3200
```

```
3290 PRINT AT 7,16; FLASH 1;"Pou
ca sorte"; FLASH 0;AT 9,15; FLAS
H 1;"a palavra e':"
3300 PAUSE 300
3310 PRINT AT 7,16;"          "
;AT 9,14;"          "
3330 PRINT AT 20,(32-nl)/2; FLAS
H 1;x$
3340 PAUSE 300
3350 PRINT AT 20,(32-nl)/2;x$
3360 GO TO 1100
3500 PRINT ,,TAB 6;"Carregar voc
abulario"
3510 PRINT ,,"Introduza nome"
3520 INPUT y$: IF LEN y$>10 THEN
 BEEP .2,24: GO TO 3520
3530 LOAD y$ DATA z$()
3540 LET ez=CODE z$(lz+1)+256*CO
DE z$(lz+2)
3550 LET pw=1
3560 LET nw=0
3570 LET pw=pw+1+CODE z$(pw)
3580 IF pw<ez THEN LET nw=nw+1:
GO TO 3570
3590 GO TO 1100
4000 PRINT AT 12,8; FLASH 1;"O p
rograma parou": PRINT AT 14,3;"P
ara recomecar introduza :"''TAB
3;"GOTO 1000 (nao apaga voc.)"''
TAB 4;"RUN (apaga vocabulario)"
4010 STOP
7999 STOP
8000 PLOT p1,p2
8010 DRAW p3,0
8020 DRAW 0,p4
8030 DRAW -p3,0
8040 DRAW 0,-p4
8050 RETURN
8100 PLOT p5,p6
8110 DRAW 0,p7
8120 RETURN
8200 PLOT p5,p8
8210 DRAW p9,0
8220 RETURN
8300 PLOT p10,p8
8310 DRAW 0,-p11
8320 RETURN
8400 CIRCLE p10,p13,p12
8410 RETURN
8500 PLOT p10,p14
8510 DRAW 0,-p15
8520 RETURN
```



```
8600 PLOT p10,p16
8610 DRAW -p17,-p18
8620 RETURN
8700 PLOT p10,p16
8710 DRAW p17,-p18
8720 RETURN
8800 PLOT p10,p19
8810 DRAW -p20,-p21
8820 RETURN
8900 PLOT p10,p19
8910 DRAW p20,-p21
8920 RETURN
```

GUARDA-REDES

Este jogo imita os jogos vídeo de bola e raqueta e pode ser jogado por uma ou duas pessoas. Vence quem primeiro marca 15 golos. O programa é bastante fácil de compreender, mas utiliza várias técnicas a que vale a pena fazer referência, incluindo um método simples para carregar em memória RAM caracteres gráficos definidos pelo utilizador, a utilização de OVER e INVERSE para obter as características de *écran* desejadas e o uso do comando IN para interrogar o teclado.

A construção dos caracteres gráficos e seu armazenamento em memória são descritos no capítulo 14 do manual de programação BASIC do *ZX Spectrum*. São reservados 168 *bytes* no topo da RAM para um total de 21 caracteres, cabendo a cada um deles oito *bytes*. A forma dos caracteres é determinada pelos números contidos em cada *byte*, e estes podem ser alterados por intermédio do comando POKE USR.

O programa «Guarda-redes» utiliza quatro caracteres especiais — o que representa a bola (letra B), o que representa a bola em «inverso» ou «negativo» (letra C), o que representa os dois guarda-redes (letra R) e o que representa os dois guarda-redes em «inverso» (letra S). Inicialmente, os números que determinam a forma destes caracteres gráficos encontram-se contidos nas declarações DATA das linhas 200 a 230 e são posteriormente transferidos para a área dos caracteres gráficos pelos ciclos das linhas 300 a 460 quando se executa o programa.

Considere-se agora o primeiro ciclo, nas linhas 300 a 340. A linha 300 guarda na variável u o valor da variável de sistema UDG, a qual indica o início da área de caracteres gráficos definíveis pelo utilizador. O primeiro carácter carregado em memória é a «bola», a qual substitui o carácter gráfico B. B é a segunda letra do alfabeto e, portanto, os valores obtidos da declaração DATA são inseridos por meio de POKE no segundo grupo de oito *bytes* na área dos caracteres gráficos. O ciclo seguinte, contido nas linhas 350 a 380, insere, também através de POKE, os oito valores seguintes nas células *(bytes)* correspondentes ao carácter gráfico C, e assim por diante.

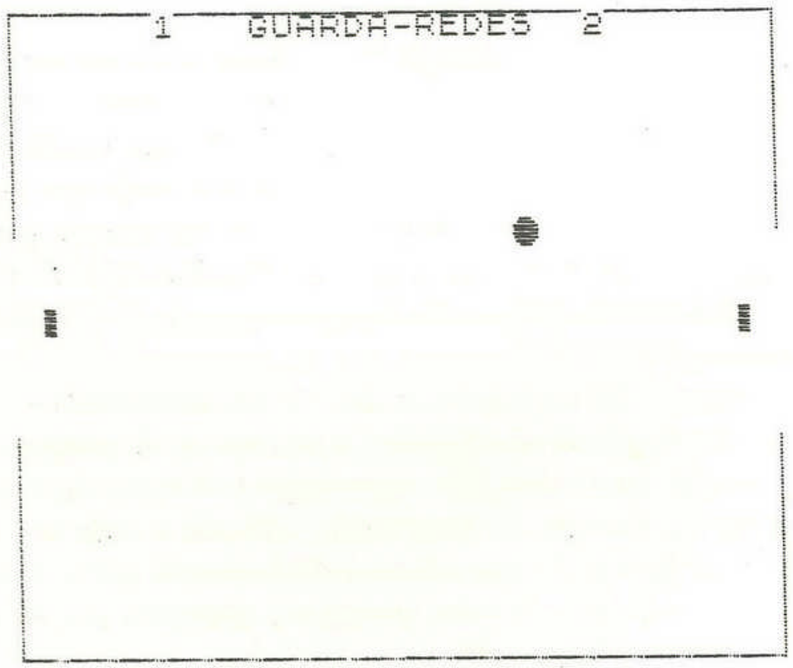


Figura 8. *O campo de futebol.*

A linha 1000 desloca a bola através da área de jogo. A declaração PRINT inclui um OVER 1, de modo a que, quando se coloca a imagem da bola numa nova posição de impressão, ela se sobrepõe mas não substitui a imagem eventualmente já nessa posição. A melhor altura de visualizar este efeito é quando a bola atinge uma parede. Os *pixels* que só coincidirem com a imagem da bola ou da parede tomam a cor da tinta (INK), mas aqueles

que coincidirem simultaneamente com as imagens da bola e da parede passam para cor de fundo ou papel (PAPER).

A linha 2000 executa a função complementar, que consiste em apagar a imagem da bola da sua posição precedente. Consegue-se este efeito imprimindo nessa posição uma imagem em «inverso» da bola (isto é: uma imagem na qual apenas os *pixels* que não façam parte da bola estejam iluminados). Emprega-se aqui de novo OVER 1, de modo a acrescentar a imagem em «inverso» à imagem da bola para formar uma posição de carácter totalmente na cor da «tinta» *com excepção dos* pixels *onde a bola tenha sido impressa sobre* (OVER) *outra imagem*. O comando INVERSE toma então a posição de carácter mencionada da cor do fundo, invertendo simplesmente as cores, apagando deste modo a imagem da bola mas deixando reaparecer qualquer imagem subjacente.

Utiliza-se uma técnica análoga para deslocar os guarda-redes. Estes são comandados, a partir do teclado, pela segunda e quarta filas de teclas. Quando accionada, qualquer das teclas da metade esquerda da segunda fila desloca para cima o guarda-redes da esquerda. De modo idêntico, a metade esquerda da fila de baixo desloca-o para baixo, enquanto que a metade direita da segunda e quarta filas comanda o guarda-redes da direita.

Com o comando IN verifica-se se as teclas mencionadas foram ou não accionadas, tal como se descreve no capítulo 23 do manual de programação BASIC do *ZX Spectrum*. Este comando permite que ambos os jogadores pressionem as teclas simultaneamente, o que não seria possível com a utilização de INKEY$, dado que INKEY$ apenas fornece um carácter como resultado se uma e uma só tecla tiver sido pressionada. Accionando-se mais do que uma tecla, INKEY$ dá um carácter nulo como resultado.

Programa 5. *«Guarda-redes».*

```
 1 REM "GUARDA-REDES"
 5 REM "●"=B "○"=C "▌"=R "▐"=S
   NOTA Introduza estas letras
        em MODO GRAFICO
10 BORDER 4: PAPER 4: INK 0: C
LS
```

```
  20 PRINT TAB 10;"GUARDA-REDES"
 100 DRAW 255,0: DRAW 0,60: PLOT
 255,116: DRAW 0,59: DRAW -255,0
: DRAW 0,-60: PLOT 0,60: DRAW 0,
-60
 110 LET t=.0005: LET n=36
 120 LET a=0: LET b=0
 200 DATA 60,126,255,255,255,255
,126,60
 210 DATA 195,129,0,0,0,0,129,19
5
 220 DATA 60,60,60,60,60,60,60,6
0
 230 DATA 195,195,195,195,195,19
5,195,195
 300 LET u=PEEK 23675+256*PEEK 2
3676
 310 FOR i=0 TO 7
 320 READ j
 330 POKE u+8+i,j
 340 NEXT i
 350 FOR i=0 TO 7
 360 READ j
 370 POKE u+16+i,j
 380 NEXT i
 390 FOR i=0 TO 7
 400 READ j
 410 POKE u+136+i,j
 420 NEXT i
 430 FOR i=0 TO 7
 440 READ j
 450 POKE u+144+i,j
 460 NEXT i
 500 LET u=2*INT (2*RND)-1: LET
v=2*INT (2*RND)-1
 510 LET c=15+(v+1)/2: LET r=INT
 (22*RND)
 520 PRINT FLASH 1;AT 0,6;a;AT 0
,24;b
 600 LET p=10: LET q=10: PRINT O
VER 1;AT p,1;"■": PRINT OVER 1 A
T q,30;"■"
 610 PAUSE 50
1000 PRINT OVER 1;AT r,c;"●"
1010 LET or=r: LET oc=c
1020 IF c=1 AND r=p THEN LET c=2
*(v<0): LET r=r+u: LET v=-v: GO
TO 1100
1030 IF c=30 AND r=q THEN LET c=
29+2*(v<0): LET r=r+u: LET v=-v:
 GO TO 1100
1040 LET r=r+u: LET c=c+v
```



```
1050 IF r<0 OR r>21 THEN BEEP t,
n: LET r=21*(r>21): LET u=-u
1060 IF (r<7 OR r>13) AND (c<0 O
R c>31) THEN BEEP t,n: LET c=31*
(c>31): LET v=-v
1100 IF IN 57342<191 THEN PRINT
OVER 1; INVERSE 1;AT q,30;"■": L
ET q=q-(q<>0): PRINT OVER 1;AT q
,30;"■"
1110 IF IN 32766<191 THEN PRINT
OVER 1; INVERSE 1;AT q,30;"■": L
ET q=q+(q<>21): PRINT OVER 1;AT
q,30;"■"
1120 IF IN 64510<191 THEN PRINT
OVER 1; INVERSE 1;AT p,1;"■": LE
T p=p-(p<>0): PRINT OVER 1;AT p,
1;"■"
1130 IF IN 65278<191 THEN PRINT
OVER 1; INVERSE 1;AT p,1;"■": LE
T p=p+(p<>21): PRINT OVER 1;AT p
,1;"■"
2000 PRINT OVER 1; INVERSE 1;AT
or,oc;" "
2010 IF c>=0 AND c<=31 THEN GO T
O 1000
3000 PRINT OVER 1; INVERSE 1;AT
p,1;"■"
3010 PRINT OVER 1; INVERSE 1;AT
q,30;"■"
3020 LET a=a+(c>31): LET b=b+(c<
0)
3030 FOR i=0 TO 8
3040 PRINT FLASH 1; INK i;AT 0,6
;a;AT 0,24;b
3050 BEEP .05,i
3060 NEXT i
3080 IF a<15 AND b<15 THEN GO TO
 500
3090 PAUSE 20: RUN
```

MÚSICA

Eis um programa destinado a ajudar a escrever música utilizando o alto-falante do *Spectrum*. O programa pode escrever ou modificar uma peça de música, apresentá-la no *écran* ou através da impressora, transpô-la para outro tom e alterar o andamento. A melodia pode, obviamente, ser tocada e os dados a ela

referentes podem ainda ser gravados separadamente em *cassette,* verificada a sua gravação e carregados de novo em memória em qualquer altura posterior.

Faz-se largo uso do comando BEEP, que se explica com alguma minúcia no capítulo 19 do manual de programação BASIC do *Spectrum.* O formato do comando é o seguinte:

BEEP duração, tom

onde a duração é o tempo em segundos e o tom, a frequência do som, é o número de meios-tons relativamente ao dó central (*middle C*, na notação inglesa; ver nota da p. 51). Ao escrever uma melodia, estamos, com efeito, a construir uma sequência satisfatória de pares de valores respeitantes à duração e tom, os quais são armazenados pelo programa num quadro bidimensional.

Quando o programa é executado, entra numa rotina inicializante que determina o tom e o andamento de referência da melodia e define um quadro destinado a conter as próprias notas (pares de valores). Uma capacidade de 500 notas para o quadro destinado a conter a melodia, um andamento de 300 a 400 pulsações por minuto e um tom de referência de dó central (seleccionada introduzindo 0) são valores típicos neste programa. Para maior comodidade, quando se passa a melodia para *cassette*, são colocados os valores da tonalidade e do andamento nos dois primeiros elementos do quadro.

```
Musica do Spectrum

        Menu

Escrever                E
Apresentar              A
Imprimir (LPrint)       I
Tocar                   T
Andamento e tom         R
Carregar (Load)         L
Gravar (Save)           S
Verificar               V
Parar                   P
```

Figura 9. *O* menu *da «Música».*

Controla-se o programa por intermédio do *menu* reproduzido na figura 9, o qual é gerado na linha 200. Ao introduzir qualquer das letras indicadas na coluna da direita, o programa entra na rotina apropriada, segundo as directrizes da linha 220.

Escrever uma melodia Introduzir E

O programa solicita o número da nota em que se pretende começar a escrever — normalmente será 1. Seguidamente, pede a duração relativa de cada nota, em número de pulsações de compasso, e o valor da sua tonalidade em número de meios-tons relativamente ao tom de referência. Assim, por exemplo, uma nota com a duração de 4 pulsações e situada 2 meios-tons acima do dó central, é introduzida na forma de 4 seguido de 2.

Se se cometer um erro, a introdução de −999 faz com que o programa volte à nota anterior. Para voltar ao *menu* introduz-se 999.

Apresentar e imprimir (LPRINT) Introduzir A ou I

Estas opções apresentam no *écran* ou através da impressora a sequência de notas que estiver em memória.

Andamento e tom Introduzir C

Quando se selecciona esta opção, o programa solicita o número de pulsações por minuto e o tom de referência relativo ao dó central. Quando os obtém, transpõe a melodia em memória de acordo com esses novos valores.

Tocar Introduzir T

Esta rotina faz o *Spectrum* tocar a melodia que se encontra em memória. Se for pressionada uma tecla enquanto a melodia está a ser tocada, o programa pára e apresenta o número da última nota tocada. Esta disposição é muito útil para encontrar os erros numa peça.

A tabela 2 apresenta as notas da conhecida canção infantil inglesa *Three Blind Mice* (Três Ratos Cegos). É necessário um espaço para o mínimo de 51 notas e sugere-se para melhor efeito que seja tocada a 500 pulsações por minuto no tom 12 relativo ao dó central.

Tabela 2. *As notas de* Three Blind Mice.

```
Apresentar melodia

Nota num.     Duracao          Tom
```


Talvez a forma mais óbvia pela qual se pudesse desenvolver este programa fosse aquela em que se introduzem as letras convencionais, sustenidos e bemóis ao escrever a peça de música. Poder-se-ia fazê-lo guardando para referência as letras, etc., num quadro de caracteres (quadro de cadeias) e comparando o que se introduzir com o conjunto de referência, de forma a identificar o número da nota. O resto do programa ficaria praticamente inalterado.

Nota: As notações inglesa, alemã e norte-americana utilizam com frequência as sete primeiras letras do alfabeto para representar as sete notas, ordenadas a partir de lá (letra A). Deste modo, o si corresponderá à letra B, o dó à letra C (como foi mencionado) e assim por diante. No caso de não ser utilizada esta notação, a modificação do programa sugerida no fim do texto continua a ser válida, bastando substituir as letras pelas iniciais (ou mesmo abreviaturas) das notas respectivas.

Programa 6. *«Música»*.

```
  1 REM "MUSICA"
 10 CLS : PRINT TAB 7;"Musica d
o Spectrum",,,"Introduza:",,"  N
um. de notas",,"  Num. de pulsac
oes p/ minuto ","  Tom relativo
ao Do' central"
 20 INPUT nn,nb,k
 30 LET nb=nb/60
 40 LET fim=0
 50 DIM b(nn+2,2)
 60 LET b(1,1)=nn
 70 LET b(1,2)=nb
 80 LET b(2,1)=k
200 CLS : PRINT TAB 7;"Musica d
o Spectrum",,,TAB 14;"Menu",,,TA
B 7;"Escrever";TAB 24;"E",TAB 7;
"Apresentar";TAB 24;"A",TAB 7;"I
mprimir(LPrint)";TAB 24;"I",TAB
7;"Tocar";TAB 24;"T",TAB 7;"Anda
```

```
mento e tom";TAB 24;"R",TAB 7;"C
arregar (Load)";TAB 24;"L",TAB 7
;"Gravar (Save)";TAB 24;"S",TAB
7;"Verificar";TAB 24;"V",TAB 7;"
Parar";TAB 24;"P"
 210 INPUT z$: GO SUB 5000
 220 GO TO 300+200*(z$="E")+700*
(z$="R")+1200*(z$="A")+1700*(z$=
"L")+2200*(z$="S")+2700*(z$="T")
+3200*(z$="V")+3700*(z$="P")+420
0*(z$="I")
 300 PAUSE 200
 310 GO TO 200
 500 PRINT ,,"Escrever/corrigir
uma melodia",,,"Introduza o nume
ro da nota em   que quer comecar
 a escrever"
 510 INPUT j: IF j<1 OR j>nn THE
N GO TO 510
 520 PRINT ,,"Introduza a duraca
o relativa e otom de cada nota."
,,,"Introduza:",,"   999 p/ termi
nar"'" -999 p/ voltar atras"
 540 PRINT ,,"Nota num.";TAB 13;
"Duracao";TAB 27;"Tom"
 550 FOR i=j+2 TO nn+2
 560 INPUT "Duracao: ";a: IF a=-
999 THEN LET i=i-1: GO TO 580
 570 IF a=999 THEN GO TO 900
 580 INPUT "Tom: ";b: IF b=-999
THEN GO TO 560
 590 LET b(i,1)=a/nb: LET b(i,2)
=k+b
 600 PRINT TAB 4;i-2;TAB 16;nb*b
(i,1);TAB 28;b(i,2)-k
 610 NEXT i
 620 PRINT ,,"Espaco totalmente
preenchido"
 630 LET fim=nn+2
 640 GO TO 300
 900 IF i>fim THEN LET fim=i-1
 910 GO TO 200
1000 PRINT ,,TAB 4;"Alterar o an
damento e o tom",,,TAB 6;"Os val
ores actuais sao:"'TAB 6;"Batid
as/minuto : ";nb*60,TAB 6;"Tom";
TAB 21;": ";k''TAB 4;"Introduza
novos valores"
1010 INPUT nc,l
1020 FOR i=3 TO fim
1030 LET b(i,1)=b(i,1)*nb*60/nc
1040 LET b(i,2)=b(i,2)-k+l
```



```
1050 NEXT i
1060 LET nb=nc/60
1070 LET k=l
1080 GO TO 60
1500 PRINT ,,TAB 7;"Apresentar m
elodia",,,"Nota num.";TAB 13;"Du
racao";TAB 27;"Tom"
1510 FOR i=3 TO fim
1520 PRINT TAB 4;i-2;TAB 16;nb*b
(i,1);TAB 28;b(i,2)-k
1530 NEXT i
1540 PRINT ,,"Pressione uma tecl
a p/ voltar ao";TAB 14;"menu"
1550 PAUSE 32768
1560 GO TO 200
2000 PRINT ,,"Carregar uma melod
ia em memoria",,,"   Introduza o
nome da melodia:"
2010 INPUT m$: IF LEN m$>10 THEN
 PRINT "Nome comprido demais. Co
rrija-o": GO TO 2010
2020 LOAD m$ DATA b()
2030 LET nn=b(1,1)
2040 LET nb=b(1,2)
2050 LET k=b(2,1)
2060 LET fim=b(2,2)
2070 PRINT ,,"   ";m$;" em memori
a"
2080 GO TO 300
2500 PRINT ,,TAB 7;"Gravar uma m
elodia",,,"   Introduza o nome da
 melodia:"
2510 INPUT m$: IF LEN m$>10 THEN
 PRINT "Nome comprido demais. Co
rrija-o": GO TO 2510
2520 LET b(2,2)=fim
2530 SAVE m$ DATA b()
2540 PRINT ,,"   ";m$;" esta gravado"
2550 GO TO 300
3000 PRINT ,,TAB 7;"Tocar uma me
lodia"
3010 PAUSE 100
3020 FOR i=3 TO fim
3030 BEEP b(i,1),b(i,2)
3040 IF INKEY$<>"" THEN GO TO 31
00
3050 NEXT i
3060 GO TO 200
3100 PRINT ,,"   Interrompida na
nota num. ";i-2''"   Deseja conti
nuar (S or N) ?"
```

```
3110 INPUT c$: IF c$<>"S" AND c$
<>"s" AND c$<>"N" AND c$<>"n" TH
EN GO TO 3110
3120 IF c$="S" OR c$="s" THEN GO
 TO 3030
3130 GO TO 300
3500 PRINT ,,TAB 5;"Verificar um
a melodia",,,"  Introduza o nome
 da melodia:"
3510 INPUT m$: IF LEN m$>10 THEN
 PRINT "Nome comprido demais. Co
rrija-o": GO TO 3510
3520 VERIFY m$ DATA b()
3530 PRINT ,,"  ";m$;" verificad
o : OK"
3540 GO TO 300
4000 PRINT AT 14,7; FLASH 1;"O p
rograma parou"
4010 PRINT AT 16,2;"Para recomec
ar introduza:"'"  GOTO 200 (nao
 apaga dados)"'TAB 5;"RUN (apag
a dados)"
4020 STOP
4500 LPRINT ,,TAB 6;"Apresentar
melodia",,,"Nota num.";TAB 13;"D
uracao";TAB 27;"Tom"
4510 FOR i=3 TO fim
4520 LPRINT "    ";i-2;TAB 16;nb
*b(i,1);TAB 28;b(i,2)-k
4530 NEXT i
4540 GO TO 300
5000 LET z$=CHR$ (CODE z$-32*(CO
DE z$>96))
5010 RETURN
```


CAPÍTULO 3

Programa de estatística e de matemática

DIAGRAMA DE DISPERSÃO

Escrevemos este programa e os dois programas seguintes, «Regressão» e «Histograma», para mostrar que o *ZX Spectrum* pode ser utilizado tanto em aplicações sérias (neste caso a análise de dados) como nas aplicações mais fúteis geralmente associadas com um computador pessoal. Elaborámos os três programas de modo a assemelharem-se quanto possível e a utilizarem a mesma estrutura de variáveis, para que os dados introduzidos num programa possam ser gravados em *cassette* e depois carregados em memória durante a execução de um dos outros dois. Considerados em conjunto, os três programas formam o núcleo de um bloco de programas-produto *(package)* para a análise estatística de dados.

O primeiro passo numa análise consiste em representar graficamente os dados de modo a permitir uma familiarização com eles e a obter uma impressão visual da sua variabilidade. É também a forma mais fácil de verificar se algum ou alguns dos valores estão incorrectos, quer por terem sido mal determinados (erros de medição, por exemplo) quer por terem sido cometidos erros ao introduzi-los no computador. O programa «Diagrama de dispersão» permite introduzir os dados por meio do teclado ou a partir de *cassette*, apresentá-los no *écran* ou através da impressora, representá-los graficamente, passá-los para *cassette* e corrigir quaisquer valores incorrectos.

Descrição geral do programa

O programa oferece a escolha entre introduzir os dados pelo teclado ou carregá-los directamente em memória a partir da *cassette*. Optando-se pela introdução através do teclado, é necessário introduzir primeiramente o número n de pontos a representar. O programa define então o quadro principal

x(2,n+1) e armazena o valor de n em x(1,1). Seguidamente somos «convidados» a introduzir n pares de valores x e y, um valor de cada vez.

Introduzidos todos os dados (ou completado o carregamento em memória), o programa imprime o *menu* (linha 1000) apresentado na figura 10 e espera que se seleccione uma opção para seguidamente «saltar» para a rotina seleccionada (linha 1050). Só a rotina destinada à representação gráfica necessita de outras explicações, posto que o resto do programa é facilmente compreensível.

```
Grafico de pontos

        Menu

Rever dados              R
Imprimir dados           I
Corrigir um valor        C
Desenhar grafico         D
Gravar (Save) dados      G
Verificar dados          V
Parar o programa         P
```

Figura 10. *O* menu *do «Diagrama de dispersão».*

Na rotina mencionada, o computador imprime o menor e o maior dos valores tanto de x como de y e pede introdução dos valores mínimo e máximo a serem utilizados em cada eixo de coordenadas. Este pormenor oferece a oportunidade de representar graficamente a totalidade ou apenas parte dos dados, à escolha. Podem igualmente escolher-se «números redondos» para os extremos do gráfico ou diagrama.

O programa elabora a representação gráfica dos dados na forma exemplificada na figura 11, omitindo quaisquer valores que se situem fora dos limites seleccionados. O número de pontos omitidos é apresentado no canto inferior esquerdo do gráfico. Cada eixo leva uma escala de cinco divisões numeradas, ficando portanto dividido em quatro intervalos. Os números da escala são truncados se excederem quatro caracteres, e o programa ajusta as suas posições de impressão no eixo dos xx de modo a que o número mais à direita (último) nesse eixo não ultrapasse o

extremo da linha de impressão e fique com os seus últimos algarismos impressos na linha seguinte.

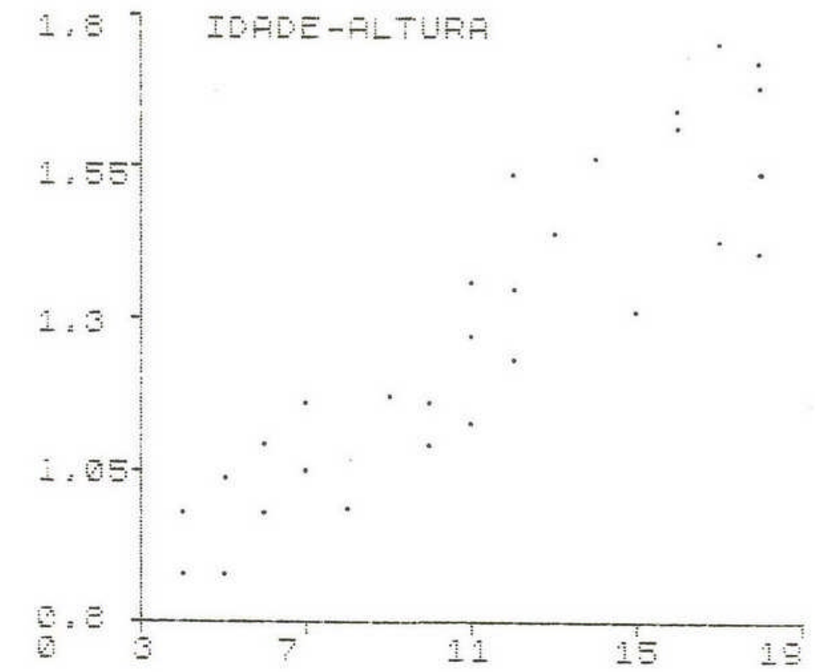


Figura 11. *Gráfico comparativo da idade com a altura de 30 jovens.*

O exemplo apresentado na figura 11 é o de um diagrama de dispersão da altura de 30 jovens em função da sua idade. Os dados encontram-se listados na tabela 3. O diagrama mostra um aumento uniforme da altura com a idade, apresentando uma tendência para ocorrer uma maior variabilidade de alturas entre os indivíduos mais velhos.

Tabela 3. *Os dados. A primeira coluna contém índices para identificar cada jovem. A segunda coluna indica a idade de cada um e a terceira coluna apresenta as alturas respectivas em metros.*

Indice	X	Y
1	7	1.05
2	12	1.54
3	7	1.16
4	10	1.16
5	11	1.13

6	4	0.88
7	14	1.56
8	10	1.09
9	6	1.09
10	12	1.23
11	16	1.61
12	17	1.75
13	13	1.44
14	5	1.04
15	6	0.98
16	15	1.31
17	6	1.09
18	16	1.64
19	4	0.98
20	9	1.17
21	18	1.68
22	5	0.88
23	8	0.99
24	18	1.54
25	11	1.27
26	18	1.41
27	11	1.36
28	12	1.35
29	17	1.43
30	18	1.72

Programa 7. *«Diagrama de dispersão»*.

```
   5 REM "DIAGRAMA DE DISPERSAO"
  10 LET t=27: LET k=5
  20 DIM b$(32)
 500 PRINT ,,TAB 4;"Diagrama de Dispersao",,,"Os dados vao ser introduzidos  pelo teclado ou por cassette";TAB 10;"(T ou C) ?"
 510 INPUT z$
 520 IF  CODE z$>96 THEN LET z$=CHR$ (CODE z$-32)
 530 IF z$<>"T" AND z$<>"C" THEN BEEP 1,12: GO TO 510
 540 GO TO 550+200*(z$="T")
 550 PRINT ,,"Introduza o nome do ficheiro em cassette : "//
 560 INPUT z$: IF LEN z$>10 THEN BEEP 1,12: GO TO 560
 570 PRINT TAB 15-LEN z$/2;z$;AT 21,0; FLASH 1;"Dados a serem carregados na RAM "
 580 LOAD z$ DATA x()
 590 PRINT AT 19,0;z$;" em memoria";TAB 31
 600 LET n=x(1,1)
 610 GO TO 1100
 750 PRINT ,,"Introduza o numero de pontos a  serem impressos no diagrama :"
 760 INPUT n: IF n<1 THEN BEEP 1,12: GO TO 760
 770 DIM x(2,n+1)
 780 PRINT "Introduza os valores de x e y",,,"Indice";TAB 14;"X";TAB 26;"Y"
 790 FOR i=2 TO n+1
 795 PRINT i-1;TAB 14;
 800 INPUT x(1,i): PRINT x(1,i);TAB 26;
 810 INPUT x(2,i): PRINT x(2,i),
 820 NEXT i
 830 LET x(1,1)=n
1000 CLS
1010 PRINT TAB 6;"Diagrama de Dispersao"
1020 PRINT ,,,,TAB 14;"Menu"
1030 PRINT ,,TAB k;"Rever dados";TAB t;"R",TAB k;"Imprimir dados";TAB t;"I",TAB k;"Corrigir um valor";TAB t;"C",TAB k;"Desenhar diagrama";TAB t;"D",TAB k;"Gravar (Save) dados";TAB t;"G",TAB k;"Verificar dados";TAB t;"V",TAB k;"Parar o programa";TAB t;"P"
1040 INPUT z$: IF CODE z$>96 THEN LET z$=CHR$ (CODE z$-32)
1050 GO TO 1100+400*(z$="R")+900*(z$="I")+1400*(z$="C")+1900*(z$="D")+3900*(z$="G")+4400*(z$="V")+4900*(z$="P")
1100 PRINT ,,"Prima uma tecla p/voltar ao menu"
1110 PAUSE 0
1120 GO TO 1000
1500 PRINT ,,TAB 10;"Rever dados",,,"Indice";TAB 14;"X";TAB 26;"Y"
1510 FOR i=2 TO n+1
1520 PRINT i-1;TAB 14;x(1,i);TAB 26;x(2,i)
1530 NEXT i
1540 GO TO 1100
2000 PRINT ,,TAB 9;"Imprimir dados",,,
2010 LPRINT ,,"Indice";TAB 14;"X
```

```
";TAB 26;"Y"
2020 FOR i=2 TO n+1
2030 LPRINT i-1;TAB 14;x(1,i);TA
B 26;x(2,i)
2040 NEXT i
2050 GO TO 1100
2500 PRINT ,,TAB 9;"Corrigir dad
os"
2510 PRINT ,,"Intr. indice do va
lor a alterar:"
2520 INPUT a: IF a<1 OR a>n THEN
 BEEP 1,12: GO TO 2520
2530 PRINT ,,"Os valores actuais
 sao :";TAB 8;x(1,a+1);TAB 20;x(
2,a+1)
2540 PRINT ,,"Introduza novos va
lores de X e Y";TAB 4;"(Um de ca
da vez)"
2550 INPUT x(1,a+1),x(2,a+1)
2560 GO TO 1100
3000 PRINT ,,TAB 5;"Desenhar dia
grama"
3010 LET xn=x(1,2)
3020 LET xx=x(1,2)
3030 LET yn=x(2,2)
3040 LET yx=x(2,2)
3050 FOR i=2 TO n+1
3060 IF xn>x(1,i) THEN LET xn=x(
1,i)
3070 IF xx<x(1,i) THEN LET xx=x(
1,i)
3080 IF yn>x(2,i) THEN LET yn=x(
2,i)
3090 IF yx<x(2,i) THEN LET yx=x(
2,i)
3100 NEXT i
3110 PRINT ,,"Menor valor de X =
 ";xn
3120 PRINT ,,"Maior valor de X =
 ";xx
3130 PRINT ,,"Menor valor de Y =
 ";yn
3140 PRINT ,,"Maior valor de Y =
 ";yx
3200 PRINT ,,"Introduza :"/"X mi
n, X max, Y min, Y max"/"relativ
os ao grafico a desenhar:",,,,
3210 PRINT "X minimo = ",: INPUT
 xn: PRINT xn
3220 PRINT "X maximo = ",: INPUT
 xx: PRINT xx
3230 PRINT "Y minimo = ",: INPUT
```



```
 yn: PRINT yn
3240 PRINT "Y maximo = ",: INPUT
 yx: PRINT yx
3300 PRINT ,,"Introduza o titulo
 do diagrama:"
3310 INPUT t$
3320 IF t$="" THEN GO TO 3400
3330 PRINT ,,"Introduza a linha
e a coluna na qual o titulo deve
s r impresso:"
3340 INPUT tl,tc
3350 IF tl<0 OR tl>21 OR tc<0 OR
 tc>31 THEN BEEP 1,12: GO TO 334
0
3400 PRINT ,,"Deseja uma copia d
o diagrama na";TAB 5;"impressora
 (S ou N) ?"
3410 INPUT c$: LET c$=CHR$ (CODE
 c$-32*(CODE c$>96))
3420 IF c$<>"S" AND c$<>"N" THEN
 BEEP 1,12: GO TO 3410
4000 CLS
4010 LET x=35: LET y=11
4020 DRAW INVERSE 1; OVER 1;x,y
4030 FOR i=1 TO 4
4040 DRAW 0,41
4050 DRAW -3,0
4060 DRAW 3,0
4070 NEXT i
4080 DRAW 0,-164
4090 DRAW -3,0
4100 DRAW 3,0
4110 DRAW 0,-3
4120 DRAW 0,3
4130 FOR i=1 TO 4
4140 DRAW 55,0
4150 DRAW 0,-3
4160 DRAW 0,3
4170 NEXT i
4180 DRAW -228,0
4200 LET a=4
4210 FOR i=xn TO 1.01*xx STEP (x
x-xn)/4
4220 LET z$=STR$ i
4230 IF LEN z$>4 THEN LET z$=z$(
 TO 4)
4240 IF a=31 THEN LET a=32-LEN z
$
4250 PRINT AT 21,a;z$
4270 LET a=a+6+(a>9)
4280 NEXT i
4400 LET a=20
```

```
4410 FOR i=yn TO 1.01*yx STEP (y
x-yn)/4
4420 LET z$=STR$ i+"    "
4430 PRINT AT a,0;z$( TO 4)
4440 LET a=a-5
4450 NEXT i
4500 LET o=0
4510 FOR i=2 TO n+1
4520 IF x(1,i)<xn OR x(1,i)>xx O
R x(2,i)<yn OR x(2,i)>yx THEN LE
T o=o+1: GO TO 4600
4530 LET x=35+220*(x(1,i)-xn)/(x
x-xn)
4540 LET y=11+164*(x(2,i)-yn)/(y
x-yn)
4550 PLOT x,y
4600 NEXT i
4610 PRINT AT 21,0;o
4700 IF t$="" THEN GO TO 4800
4710 PRINT AT tl,tc;t$
4800 IF c$="S" THEN COPY
4900 PRINT #0;"Prima uma tecla p
/voltar ao menu": PAUSE 0
4910 GO TO 1000
5000 PRINT AT 13,9;"Gravar dados
";AT 15,0;"Introduza o nome do f
icheiro :"
5010 INPUT z$: IF LEN z$>10 THEN
 BEEP 1,12: GO TO 5010
5020 PRINT AT 15,0;b$;AT 15,15-L
EN z$/2;z$;AT 21,4; FLASH 1;"Gra
vacao a ser efectuada"
5030 SAVE z$ DATA x()
5040 PRINT AT 21,0;"Gravacao de
";z$;" concluida";TAB 30
5050 GO TO 1100
5500 PRINT AT 13,8;"Verificar da
dos";AT 15,0;"Introduza o nome d
o ficheiro:"
5510 INPUT z$: IF LEN z$>10 THEN
 BEEP 1,12: GO TO 5510
5520 PRINT AT 15,0;b$;AT 15,15-L
EN z$/2;z$;AT 21,2; FLASH 1;"Ver
ificacao a ser efectuada"
5530 VERIFY z$ DATA x()
5540 PRINT AT 19,0;z$;" verifica
do : OK";TAB 30
5550 GO TO 1100
6000 PRINT AT 13,8; FLASH 1;"O p
rograma parou"
6010 PRINT AT 15,3;"Para recomec
ar introduza :"''TAB 3;"GOTO 100
0 (nao apaga dados)"''TAB 7;"RUN
 (apaga dados)"
6020 STOP
```


REGRESSÃO

Este é o segundo de três programas para a análise estatística de dados, sendo os outros dois «Diagrama de dispersão» e «Histograma». Este programa ajusta uma recta a pares de valores medidos x e y pelo método dos mínimos quadrados e imprime os coeficientes da recta de ajustamento, os erros estimados dos coeficientes e uma estimativa do erro na recta de ajustamento (erro padrão da estimativa). O programa calcula também os valores estimados, ou sejam, os valores de y obtidos por substituição dos valores observados de x na equação da recta de ajustamento, e os resíduos (diferenças entre os valores observados de y e os valores estimados). Os valores estimados e os resíduos podem ser passados para *cassette* com o mesmo formato dos dados originais, de modo a serem analisados através dos programas «Diagrama de dispersão» e «Histograma».

Este programa pede que se carreguem em memória (LOAD) os dados a analisar (se eles se encontrarem em *cassette*) ou que se introduzam através do teclado. Feito isto, imprime o *menu* apresentado na figura 12. As opções disponíveis são as mesmas

```
                Regressao

                  Menu

Rever dados                    R
Imprimir dados                 I
Corrigir um valor              C
Ajustar regressao              A
Gravar dados                   G
Verificar dados                V
Parar o programa               P
```

Figura 12. *O* menu *de «Regressão».*

do programa «Diagrama de dispersão», com excepção da opção «Ajustamento da regressão». Deste modo, pode fazer-se com que os dados sejam apresentados no *écran* ou através da impressora, corrigir valores incorrectos, gravar os dados em *cassette* e verificar a gravação.

Seleccionando a opção «Ajustamento da regressão» o programa calcula a média dos valores de x e a média dos valores de y (xb e yb), a soma dos quadrados dos valores de x e a soma dos quadrados dos valores de y (sxx e syy) e a soma dos produtos cruzados dos valores de x e de y (sxy). A forma mais condensada das equações de cada um destes elementos é dada seguidamente:

$$xb = \sum_{i=1}^{n} x_i/n$$

$$yb = \sum_{i=1}^{n} y_i/n$$

$$sxx = \sum_{i=1}^{n} (x_i - xb) * (x_i - xb)$$

$$syy = \sum_{i=1}^{n} (y_i - yb) * (y_i - yb)$$

$$sxy = \sum_{i=1}^{n} (x_i - xb) * (y_i - yb)$$

onde n representa tamanho da amostra, e $\sum_{i=1}^{n}$ significa «somatório dos valores da expressão (escrita a seguir) quando i varia de 1 até n», ou seja, a soma dos n valores que a expressão toma ao variar i de 1 até n.

É fácil de ver que os valores de sxx, syy e sxy só podem ser

determinados depois de xb e yb terem sido calculados. Deste modo, para aplicar directamente estas equações, na forma em que elas se apresentam, o programa teria de efectuar duas passagens através do quadro de dados. Na primeira passagem, somar-se-iam os valores de x e os valores de y, de modo a que, no fim da passagem, dividindo cada uma das duas somas por n, se obtivessem xb e yb. Estes valores seriam então substituídos nas equações relativas a sxx, sxy e syy, fazendo-se seguidamente uma segunda passagem pelos dados para calcular os somatórios.

Efectuar duas passagens através dos dados é um processo moroso e, portanto, o programa utiliza uma forma adaptada das equações referidas, de modo a que todos os cinco elementos (xb, yb, sxx, sxy e syy) sejam calculados numa única passagem. As novas formas das equações encontram-se nas linhas 3060 a 3100.

O programa utiliza os cinco elementos para calcular a equação da recta de regressão (linhas 3150 a 3450). Um exemplo de entrada é apresentado na figura 13. Os resultados foram gerados pelo mesmo conjunto de dados do «Diagrama de dispersão», nomeadamente a altura, em metros, e a idade, em anos, de 30 jovens.

```
        Ord. na o        Declive

Coef    0.72059301       .050333957
Err     .049844433       .0044082279
T       14.45684         11.418184

        Linha            Residuo
GL      1                28
SO      1.6234715        0.3486852
SMO     1.6234715        .012452329
F       130.37493
```

Figura 13. *Um exemplo de* output *do programa «Regressão».*

A figura 13 mostra que a equação de regressão pelos mínimos quadrados que relaciona a altura com a idade é a seguinte:

$$\text{altura} = 0.72 + 0.05 * \text{idade}$$

Além disso, os erros relativos aos valores da ordenada na origem e do declive são 0.05 e 0.0044, respectivamente, e os valores da distribuição T de Student são ambos elevados, o que indica que tanto a ordenada na origem como o declive apresentam valores significativamente diferentes de zero. A metade inferior da figura 13 é uma análise da tabela de variância. O significado de cada uma das abreviaturas aí apresentadas é o seguinte:

GL — Graus de liberdade
SQ — Soma de quadrados
SMQ — Soma média de quadrados
F — Distribuição F de Snedecor

O valor de F apresentado na figura é muito grande, o que significa que existe uma forte relação entre a altura e a idade.

Programa 8. *«Regressão».*

```
    5 REM "REGRESSAO"
   10 LET t=30
  500 PRINT ,,TAB 11;"Regressao",
,,"Os dados vao ser introduzidos
    pelo teclado ou carregados de
    uma cassete  (T ou C) ?"
  510 INPUT z$
  520 IF  CODE z$>96 THEN LET z$=
CHR$ (CODE z$-32)
  530 IF z$<>"T" AND z$<>"C" THEN
 BEEP 1,12: GO TO 510
  540 GO TO 550+200*(z$="T")
  550 PRINT ,,"Introduza o nome d
o ficheiro em cassette:"
  560 INPUT z$: IF LEN z$>10 THEN
 BEEP 1,12: GO TO 560
  570 PRINT AT 21,2; FLASH 1;"Car
regamento a ser efectuado"
  580 LOAD z$ DATA x()
  590 PRINT AT 19,0;z$;" em memor
ia";TAB 30
  600 LET n=x(1,1)
  610 GO TO 1100
  750 PRINT ,,"Intr. numero de pa
res de dados:"
  760 INPUT n: IF n<1 THEN BEEP 1
,12: GO TO 760
  770 DIM x(2,n+1)
  780 PRINT "Introduza os valores
 de x e de yum de cada vez:"'"I
ndice";TAB 14;"X";TAB 26;"Y"
  790 FOR i=2 TO n+1
  795 PRINT i-1;TAB 14;
  800 INPUT x(1,i): PRINT x(1,i);
TAB 26;
  810 INPUT x(2,i): PRINT x(2,i),
  820 NEXT i
  830 LET x(1,1)=n
 1000 CLS
 1010 PRINT TAB 11;"Regressao"
 1020 PRINT ,,,,TAB 14;"Menu"
 1030 PRINT ,,"Rever dados";TAB t
;"R","Imprimir dados";TAB t;"I",
"Corrigir um valor";TAB t;"C","A
justar regressao";TAB t;"A","Gra
var dados";TAB t;"G","Verificar
dados";TAB t;"V","Parar o progra
ma";TAB t;"P"
 1040 INPUT z$: IF CODE z$>96 THE
N LET z$=CHR$ (CODE z$-32)
 1050 GO TO 1100+400*(z$="R")+900
*(z$="I")+1400*(z$="C")+1900*(z$
="A")+3900*(z$="G")+4400*(z$="V"
)+4900*(z$="P")
 1100 PRINT ,,"Prima uma tecla p/
Voltar ao menu"
 1110 PAUSE 0
 1120 GO TO 1000
 1500 PRINT ,,TAB 10;"Rever dados
",,,"Indice";TAB 14;"X";TAB 26;"
Y"
 1510 FOR i=2 TO n+1
 1520 PRINT i-1;TAB 14;x(1,i);TAB
 26;x(2,i)
 1530 NEXT i
 1540 GO TO 1100
 2000 PRINT ,,TAB 10;"Imprimir da
dos",,,
 2010 LPRINT ,,"Indice";TAB 14;"X
";TAB 26;"Y"
 2020 FOR i=2 TO n+1
 2030 LPRINT i-1;TAB 14;x(1,i);TA
B 26;x(2,i)
 2040 NEXT i
 2050 GO TO 1100
```

```
2500 PRINT ,,TAB 9;"Corrigir dad
os"
2510 PRINT ,,"Intr. indice do va
lor a alterar:"
2520 INPUT a: IF a<1 OR a>n THEN
 BEEP 1,12: GO TO 2520
2530 PRINT ,,"Os valores actuais
 sao :"''TAB 8;x(1,a+1);TAB 20;x
(2,a+1)
2540 PRINT ,,"Introduza novos va
lores de x e yum de cada vez :"'
2550 INPUT x(1,a+1),x(2,a+1)
2560 GO TO 1100
3000 PRINT ,,TAB 8;"Ajustar regr
essao"
3010 PRINT ,,"Deseja copia impre
ssa (S ou N) ?"
3020 INPUT y$: LET y$=CHR$ (CODE
 y$-32*(y$>"£")): IF y$<>"S" AND
 y$<>"N" THEN BEEP .5,24: GO TO
3020
3030 IF y$="S" THEN CLS
3040 LET xb=0: LET yb=0: LET sxx
=0: LET syy=0: LET sxy=0
3050 FOR i=1 TO n
3060 LET sxx=sxx+(i-1)*(x(1,i+1)
-xb)*(x(1,i+1)-xb)/i
3070 LET syy=syy+(i-1)*(x(2,i+1)
-yb)*(x(2,i+1)-yb)/i
3080 LET sxy=sxy+(i-1)*(x(1,i+1)
-xb)*(x(2,i+1)-yb)/i
3090 LET xb=((i-1)*xb+x(1,i+1))/
i
3100 LET yb=((i-1)*yb+x(2,i+1))/
i
3110 NEXT i
3150 LET b=sxy/sxx
3160 LET seb=((syy*sxx-sxy*sxy)/
(n-2))↑.5/sxx
3170 LET a=yb-b*xb
3180 LET sea=(syy/(n*(n-1))+xb*s
eb*seb)↑.5
3300 LET s1=5: LET s2=19
3310 PRINT ,,TAB s1;"Ord. na o."
;TAB s2;"Declive"
3320 PRINT ,,,,"Coef ";a;TAB s2;
b
3330 PRINT ,,"Err  ";sea;TAB s2;
seb
3340 PRINT ,,"T";TAB s1;a/sea;TA
B s2;b/seb
3400 PRINT ,,,,,,TAB s1;"Linha";
```



```
TAB s2;"Residuo"
3410 PRINT ,,"GL";TAB s1;"1";TAB
 s2;n-2
3420 LET smq=b*b*sxx: LET rsq=sy
y-smq
3430 PRINT ,,"SQ";TAB s1;smq;TAB
 s2;rsq
3440 PRINT ,,"SMQ";TAB s1;smq;TA
B s2;rsq/(n-2)
3450 PRINT ,,"F";TAB s1;smq*(n-2
)/rsq
3460 IF y$="S" THEN COPY : CLS
3500 PRINT ,,"Deseja gravar os r
esiduos e os  valores esperados
(S ou N) ?"
3510 INPUT y$: LET y$=CHR$ (CODE
 y$-32*(y$>"£")): IF y$<>"S" AND
 y$<>"N" THEN BEEP .5,24: GO TO
3020
3520 IF y$="N" THEN GO TO 1100
3530 DIM y(2,n+1)
3540 FOR i=1 TO n+1
3550 LET y(1,i)=x(1,i): LET y(2,
i)=x(2,i)
3560 NEXT i
3570 FOR i=1 TO n
3580 LET x(2,i+1)=a+b*y(1,i+1)
3590 LET x(1,i+1)=y(2,i+1)-x(2,i
+1)
3595 NEXT i
3600 PRINT ,,"Introduza o nome"
3610 INPUT z$: IF LEN z$>10 THEN
 BEEP 1,12: GO TO 3610
3620 PRINT AT 21,0;"     ";AT 21,
4; FLASH 1;"Gravacao a ser efect
uada"
3630 SAVE z$ DATA x()
3640 PRINT AT 21,0;z$;" encontra
-se gravado";TAB 30
3650 PRINT ,,"Rebobine a fita pa
ra permitir  que a verificacao
se efectue  ",,,"Depois, prima
uma tecla "
3660 PAUSE 0
3670 PRINT AT 21,0;"  ";AT 21,2;
 FLASH 1;"Verificacao a ser efec
tuada"
3680 VERIFY z$ DATA x()
3690 PRINT AT 19,0;z$;" verifica
do: OK";TAB 30
3700 FOR i=1 TO n+1
3710 LET x(1,i)=y(1,i): LET x(2,
```

```
i)=y(2,i)
3720 NEXT i
3800 GO TO 1100
5000 PRINT ,,TAB 10;"Gravar dado
s",,,"Introduza o nome:"
5010 INPUT z$: IF LEN z$>10 THEN
 BEEP 1,12: GO TO 5010
5020 PRINT AT 21,4; FLASH 1;"Gra
vacao a ser efectuada"
5030 SAVE z$ DATA x()
5040 PRINT AT 21,0;z$;" encontra
-se gravado";TAB 30
5050 GO TO 1100
5500 PRINT ,,TAB 8;"Verificar da
dos",,,"Introduza o nome:"
5510 INPUT z$: IF LEN z$>10 THEN
 BEEP 1,12: GO TO 5510
5520 PRINT AT 21,2; FLASH 1;"Ver
ificacao a ser efectuada"
5530 VERIFY z$ DATA x()
5540 PRINT AT 19,0;z$;" verifica
do: OK";TAB 30
5550 GO TO 1100
6000 PRINT ,,"O programa parou"
6010 STOP
```

HISTOGRAMA

«Histograma» é o terceiro dos programas relacionados com a análise estatística de dados, sendo os dois outros o «Diagrama de dispersão» e a «Regressão». Os três programas foram elaborados para poderem trabalhar em conjunto, de forma a cada um deles poder efectuar um aspecto diferente de uma análise de um determinado conjunto de dados. Característica importante destes programas: os dados introduzidos num deles através do teclado podem ser gravados em *cassette* e posteriormente carregados em memória durante a execução de qualquer dos outros programas, evitando assim o trabalho de os introduzir novamente pelo teclado.

Uma diferença importante entre o «Histograma» e outros dois programas é que apenas se analisa uma variável e não duas. Assim, quando se transferem os dados de outro programa para o

«Histograma», este programa pede a selecção de uma das variáveis para análise. A variável não seleccionada é posta de parte e pode ser requisitada mais tarde pela escolha da opção apropriada no *menu* principal. Ao introduzir os dados no «Histograma» através do teclado, presume-se que apenas se empregaram os valores de uma variável e, portanto, não é aconselhável transferir tais dados para análise através dos outros programas.

A figura 14 apresenta as opções disponíveis no «Histograma». Só descrevemos aqui a opção «Desenhar histograma» devido ao facto de as restantes opções serem auto-explicativas.

```
                Histograma

                   Menu

Consultar dados                          C
Imprimir dados                           I
Alterar um valor                         A
Desenhar histograma                      D
Parar o programa                         P
Seleccione outra variavel                S
```

Figura 14. Menu *de «Histograma»*.

Ao seleccionar a opção «Desenhar histograma», o programa imprime os valores menor e maior da variável e pede a indicação de um limite superior e um limite inferior para o histograma e o número de barras necessário. Deste modo, há a oportunidade de estudar a totalidade ou apenas parte dos dados e podem escolher-se «números redondos» para os limites de cada barra do histograma. O programa arredonda o número de barras para o múltiplo de quatro mais próximo devido ao facto de colocar cinco números de escala sobre o eixo vertical do histograma, determinando assim quatro intervalos.

A figura 15 apresenta o *output* do programa quando este é aplicado aos resíduos da regressão da altura e idade de 30 jovens. A distribuição é satisfatória, apesar de existir um número surpreendentemente grande entre 0.05 e 0.1.

```
Limite inf.          Contagem

-0.15                1
-0.1                 4
-.05                 4
0                    5
.05                  1
0.1                  10
0.15                 2
0.2                  1

1 menor que o limite inferior
1 maior que o limite superior

Prima uma tecla para continuar
```

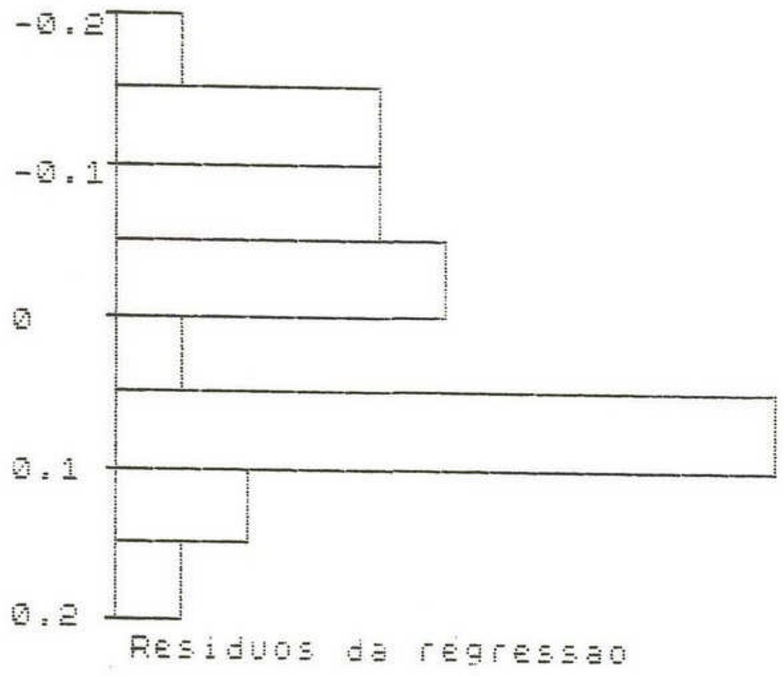


Figura 15. *Um exemplo de* output *de «Histograma».*

```
Indice               Valor
                     -.022930712
                     0.21539995
                     .087059288
                     -.063932584
                     -0.14426654
                     -.041928839
                     0.13473159
                     -0.13393258
                     .057403246
                     -.094600499
                     .084063671
                     0.17372971
                     .065065544
                     .067737204
                     -.042595754
                     -0.16560237
                     .057403246
                     0.11406367
                     .058071161
                     -.00359862G7
                     .053395756
                     -.092262797
                     -0.13326467
                     -.086604244
                     -.004266554l5
                     -0.21660424
                     .085733458
                     .025399501
                     -0.14627029
                     .093395756

Residuos da regressao
```

Figura 15-A. Input *para o exemplo da figura 15.*

Programa 9. *«Histograma».*

```
  5 REM "HISTOGRAMA"
 10 LET t=30
500 PRINT ,,TAB 10;"Histograma"
,,,"Os dados vao ser introduzid
s   pelo teclado ou carregados d
e   uma cassette (T ou C) ?"
510 INPUT y$
520 IF  CODE y$>96 THEN LET y$=
CHR$ (CODE y$-32)
530 IF y$<>"T" AND y$<>"C" THEN
BEEP 1,12: GO TO 510
540 GO TO 550+200*(y$="T")
550 PRINT ,,"Introduza o nome d
o ficheiro em cassette:"
560 INPUT z$: IF LEN z$>10 THEN
BEEP 1,12: GO TO 560
570 PRINT AT 21,2; FLASH 1;"Car
```

```
ramento a ser efectuado"
  580 LOAD z$ DATA x()
  590 PRINT AT 19,0;z$;" em memor
ia";TAB 30
  600 LET n=x(1,1)
  610 PRINT ,,"Qual e' a variavel
 (X ou Y) ?"
  620 INPUT z$: LET z$=CHR$ (CODE
 z$-32*(z$>"£")): IF z$<>"X" AND
 z$<>"Y" THEN BEEP .5,24: GO TO
620
  630 LET w=1+(z$="Y")
  640 GO TO 1000
  750 PRINT ,,"Introduza o numero
 de dados:"
  760 INPUT n: IF n<1 THEN BEEP 1
,12: GO TO 760
  770 DIM x(2,n+1)
  780 PRINT "Introduza os valores
 a vez",,,"Indice","Valor"
  790 FOR i=2 TO n+1
  800 PRINT i-1,: INPUT x(1,i): P
RINT x(1,i)
  810 NEXT i
  820 LET x(1,1)=n
  830 LET w=1
1000 CLS
1010 PRINT TAB 11;"Histograma"
1020 PRINT ,,,,TAB 14;"Menu"
1030 PRINT ,,"Consultar dados";T
AB t;"C","Imprimir dados";TAB t;
"I","Alterar um valor";TAB t;"A"
,"Desenhar histograma";TAB t;"D"
,"Parar o programa";TAB t;"P"
1035 IF y$="C" THEN PRINT "Selec
cione outra variavel";TAB t;"S"
1040 INPUT z$: IF CODE z$>96 THE
N LET z$=CHR$ (CODE z$-32)
1050 GO TO 1060+440*(z$="C")+940
*(z$="I")+1440*(z$="A")+1940*(z$
="D")+4940*(z$="P")
1060 IF z$="S" AND y$="C" THEN L
ET w=2-(w=2): PRINT ,,"Foi selec
cionada outra variavel"
1100 PRINT ,,"Prima uma tecla p/
voltar ao menu"
1110 PAUSE 0
1120 GO TO 1000
1500 PRINT ,,TAB 8;"Consultar da
dos",,,"Indice","Valor"
1510 FOR i=2 TO n+1
1520 PRINT i-1,x(w,i)
```



```
1530 NEXT i
1540 GO TO 1100
2000 PRINT ,,TAB 9;"Imprimir dad
os",,,
2010 LPRINT ,,"Indice","Valor"
2020 FOR i=2 TO n+1
2030 LPRINT i-1,x(w,i)
2040 NEXT i
2050 GO TO 1100
2500 PRINT ,,TAB 10;"Alterar dad
os"
2510 PRINT ,,"Intr. indice do va
lor a alterar:"
2520 INPUT a: IF a<1 OR a>n THEN
 BEEP 1,12: GO TO 2520
2530 PRINT ,,"Valor actual = ";x
(w,a+1)
2540 PRINT ,,"Intr. novo valor:"
2550 INPUT x(w,a+1)
2560 GO TO 1100
3000 PRINT ,,TAB 10;"Histograma"
3010 LET xn=x(w,2)
3020 LET xx=x(w,2)
3030 FOR i=2 TO n+1
3040 IF xn>x(w,i) THEN LET xn=x(
w,i)
3050 IF xx<x(w,i) THEN LET xx=x(
w,i)
3060 NEXT i
3100 PRINT ,,"Menor valor = ";xn
3110 PRINT ,,"Maior valor = ";xx
3120 PRINT ,,"Introduza os limit
es minimo e   maximo do histogra
ma:"
3130 INPUT xn,xx: IF xn>=xx THEN
 BEEP .5,24: GO TO 3120
3140 PRINT ,,"Introduza o numero
 de barras de-sejado (um multipl
o de quatro):"
3150 INPUT nb: IF nb<1 THEN LET
nb=1
3160 LET nb=4*INT ((nb+3)/4)
3170 LET xs=(xx-xn)/nb
3200 PRINT ,,"Introduza titulo d
o histograma:"
3210 INPUT t$
3220 IF t$="" THEN GO TO 3280
3230 PRINT ,,"Introduza a linha
e a coluna em que o titulo deve
ser impresso:"
3240 INPUT tl,tc
3250 IF tl<0 OR tl>21 OR tc<0 OR
```

```
 tc>31 THEN BEEP .5,24: GO TO 32
40
3280 PRINT ,,"Deseja copia impre
ssa (S ou N)?"
3290 INPUT c$: LET c$=CHR$ (CODE
 c$-32*(c$>"ŧ")): IF c$<>"S" AND
 c$<>"N" THEN BEEP .5,24: GO TO
3290
3300 PRINT ,,"Limite inf.","Cont
agem",,,: IF c$="S" THEN LPRINT
"Limite inferior","Contagem",,,
3310 DIM b(nb)
3320 LET cl=0: LET ch=0
3330 FOR j=2 TO n+1
3340 LET i=INT ((x(w,j)-xn)/xs)+
1
3350 IF i<1 THEN LET cl=cl+1: GO
 TO 3380
3360 IF i>nb THEN LET ch=ch+1: G
O TO 3380
3370 LET b(i)=b(i)+1
3380 NEXT j
3390 FOR i=1 TO nb
3400 PRINT xn+i*xs,b(i): IF c$="
Y" THEN LPRINT xn+i*xs,b(i)
3410 NEXT i
3420 IF c$="Y" THEN LPRINT
3430 IF ch*cl=0 THEN GO TO 3460
3440 PRINT ,,cl;" menor"+("es" A
ND cl>1)+" que o limite inferior
",ch;" maior"+("es" AND ch>1)+"
que o limite superior"
3450 IF c$="S" THEN LPRINT cl;"
menor"+("es" AND cl>1)+" que o l
imite inferior",ch;" maior"+("es
" AND ch>1)+" que o limite super
ior",,,
3460 PRINT ,,"Prima uma tecla pa
ra continuar"
3470 PAUSE 0
3500 CLS : IF c$="S" THEN LPRINT
 : LPRINT
3510 LET x=35: LET y=11
3520 PLOT x,y
3530 DRAW -3,0
3540 DRAW 3,0
3550 FOR i=1 TO 4
3560 DRAW 0,41
3570 DRAW -3,0
3580 DRAW 3,0
3590 NEXT i
3700 LET a=0
```



```
3710 FOR i=xn TO 1.01*xx STEP (x
x-xn)/4
3720 LET z$=STR$ i+"     "
3730 PRINT AT a,0;z$( TO 4)
3740 LET a=a+5
3750 NEXT i
4000 LET mnb=0
4010 FOR i=1 TO nb
4020 IF mnb<b(i) THEN LET mnb=b(
i)
4030 NEXT i
4050 PLOT x,175
4060 FOR i=1 TO nb
4070 LET l=(255-x)*b(i)/mnb
4080 LET d=INT (164*i/nb)-INT (1
64*(i-1)/nb)
4090 DRAW l,0
4100 DRAW 0,-d
4110 DRAW -l,0
4120 NEXT i
4200 IF t$="" THEN GO TO 4220
4210 PRINT AT tl,tc;t$
4220 IF c$="S" THEN COPY
4230 PRINT #0;"Prima uma tecla p
/voltar ao menu"
4300 PAUSE 0
4310 GO TO 1000
6000 PRINT ,,"O programa parou"
6010 STOP
```

GRÁFICO DE FUNÇÕES

O programa «Gráfico de funções» elabora o traçado do gráfico de uma função por nós introduzida. Podemos seleccionar o intervalo em que queremos que a função seja representada graficamente e também obter cópias do gráfico na impressora. Apresenta-se um exemplo de *output* do programa na figura 16.

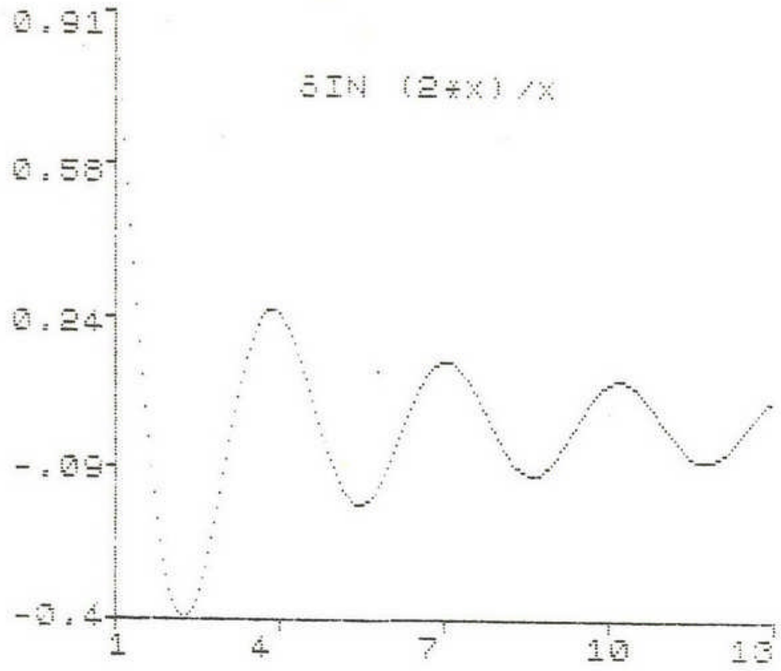


Figura 16. *Gráfico da função sin (2*x)/x no intervalo [1,13].*

Quando o programa é executado, pede a introdução da função em x a representar graficamente. O programa não verifica a sintaxe da função e, portanto, deve ter-se o cuidado de verificar se não existem quaisquer caracteres incorrectos, se os parênteses têm todos par, etc. Introduzida a função, devem introduzir-se os valores mínimo e máximo de x do intervalo desejado.

O programa pede também a introdução do valor das assímptotas (se existirem). Uma assímptota é um valor de x para o qual a função toma um valor de mais ou menos infinito. Por exemplo, a equação y=log x tem um assímptota em x=0, pois log 0 é menos infinito (ou infinito com sinal negativo). Analogamente, a equação y=tan x possui assímptotas em $x=\pi/2$, $3*\pi/2$, $5*\pi/2$, etc. Se a função introduzida tiver uma ou mais assímptotas, o programa solicita a indicação de uma tolerância para o cálculo, de modo a evitar calcular a equação para valores de x que estejam tão próximos de um valor assimptótico como a tolerância está de zero. Por outras palavras, se existir uma assímptota em x=1 e a

tolerância for 0.01, o programa não irá calcular o valor da função para valores de x entre 0.99 e 1.01.

O programa calcula a função para 221 valores de x no intervalo seleccionado, escolhe um intervalo adequado para o eixo dos y e desenha os eixos. As rotinas que imprimem os valores de escala de cada eixo (linhas 2200 a 2450) executam cinco vezes um ciclo, uma vez para cada número de escala. O valor máximo do ciclo é fixado ligeiramente acima do máximo necessário na realidade para assegurar que os erros de arredondamento acumulados não façam com que o ciclo termine logo após a quarta execução, em vez de ser depois da quinta. A rotina seguinte demonstra o que pode acontecer, no caso contrário:

```
10 INPUT j
20 FOR i=0 TO j STEP j/4
30 PRINT i
40 NEXT i
```

Se o valor de j for fixado em 1.01, a rotina imprimirá

0, 0.2525, 0.505, 0.7575, 1.01.

Se, em vez disso, o valor de j for fixado em 1.02, a rotina irá imprimir:

0, 0.255, 0.51, 0.765

No segundo caso, apenas são impressos quatro valores, devido ao facto de se terem acumulado erros de arredondamento à medida que o valor de STEP (incremento) era adicionado à variável de controle de ciclo i em cada «passo» (STEP).

Depois de ter traçado o gráfico, o programa pede a introdução de coordenadas para a declaração PRINT AT, que imprime o «título» da função no gráfico (linha 2620). OVER 1 é utilizada de forma a que, se não ficarmos satisfeitos com a posição que escolhemos para o título, este pode ser apagado pela simples repetição da sua impressão, utilizando novamente OVER 1 (linha 2650).

Programa 10. *«Gráfico de funções».*

```
  50 REM "GRAFICO DE FUNCOES"
 100 PRINT TAB 7;"Grafico de fun
coes",,,"Introduza f(x) a ser de
senhada:"
 110 INPUT f$: PRINT ,,"A funcao
 e' ";f$
 120 PRINT ,,"Introduza os valor
es minimo e   maximo entre os qu
ais a funcao  deve ser represent
ada:",,,
 130 PRINT "X minimo = ",: INPUT
 xn: PRINT xn
 140 PRINT "X maximo = ",: INPUT
 xx: PRINT xx
 150 PRINT ,,"Introduza numero d
e assimptotas:"
 160 INPUT as: IF as<0 THEN BEEP
 1,12: GO TO 160
 170 IF as=0 THEN GO TO 300
 180 DIM a(as)
 190 PRINT ,,"Introduza,um de ca
da vez, os va-lores de x em cada
 assimptota:"
 200 FOR i=1 TO as
 210 INPUT a(i)
 220 NEXT i
 230 PRINT ,,"Introduza a tolera
ncia nas      assimptotas (p.ex.
: .01) :"
 240 INPUT tol: IF tol<0 THEN BE
EP 1,12: GO TO 240
 300 PRINT AT 21,2; FLASH 1;"Cal
culos a serem efectuados"
 310 DIM x(221): DIM y(221)
 320 LET i=1
 330 FOR x=xn TO xx STEP (xx-xn)
/220
 340 IF as=0 THEN GO TO 400
 350 FOR j=1 TO as
 360 IF ABS (x-a(j))<tol THEN GO
 TO 430
 370 NEXT j
 400 LET x(i)=x
 410 LET y(i)=VAL f$
 420 LET i=i+1
 430 NEXT x
 440 LET n=i-1
 500 LET yn=y(1)
 510 LET yx=y(1)
```



```
 520 FOR i=1 TO n
 530 IF yn>y(i) THEN LET yn=y(i)
 540 IF yx<y(i) THEN LET yx=y(i)
 550 NEXT i
 560 LET yn=yn+.99
 570 LET yx=yx+1.01
 580 PRINT AT 21,0,,
1000 PRINT ,,"Deseja uma copia d
o grafico na  impressora (S ou N
) ?"
1010 INPUT c$: LET c$=CHR$ (CODE
 c$-32*(CODE c$>96))
1020 IF c$<>"S" AND c$<>"N" THEN
 BEEP 1,12: GO TO 1010
2000 CLS
2010 LET x=35: LET y=11
2020 DRAW INVERSE 1; OVER 1;x,y
2030 FOR i=1 TO 4
2040 DRAW 0,41
2050 DRAW -3,0
2060 DRAW 3,0
2070 NEXT i
2080 DRAW 0,-164
2090 DRAW -3,0
2100 DRAW 3,0
2110 DRAW 0,-3
2120 DRAW 0,3
2130 FOR i=1 TO 4
2140 DRAW 55,0
2150 DRAW 0,-3
2160 DRAW 0,3
2170 NEXT i
2180 DRAW -228,0
2200 LET a=4
2210 FOR i=xn TO xx STEP (xx-xn)
/4
2220 LET z$=STR$ i
2230 IF LEN z$>4 THEN LET z$=z$(
 TO 4)
2240 IF a=31 THEN LET a=32-LEN z
$
2250 PRINT AT 21,a;z$
2260 LET a=a+6+(a>9)
2270 NEXT i
2400 LET a=20
2410 FOR i=yn TO 1.01*yx STEP (y
x-yn)/4
2420 LET z$=STR$ i+"   "
2430 PRINT AT a,0;z$( TO 4)
2440 LET a=a-5
2450 NEXT i
2500 FOR i=1 TO n
```

```
2510 LET x=35+220*(x(i)-xn)/(xx-
xn)
2520 LET y=12+163*(y(i)-yn)/(yx-
yn)
2530 PLOT x,y
2540 NEXT i
2600 INPUT "Intr. as coord. do t
itulo ";x;" ";y
2610 IF x<0 OR x>21 OR y<0 OR y>
31 THEN BEEP 1,12: GO TO 2610
2620 PRINT OVER 1;AT x,y;f$
2630 INPUT "Esta' correcto (S ou
 N) ? ";z$
2640 LET z$=CHR$ (CODE z$-32*(z$
>"f")): IF z$<>"S" AND z$<>"N" T
HEN BEEP 1,12: GO TO 2630
2650 IF z$="N" THEN PRINT OVER 1
;AT x,y;f$: GO TO 2600
2660 IF c$="S" THEN COPY
```


Programas utilitários

RENUMERADOR DE LINHAS

Uma característica da linguagem de programação BASIC é a utilização de números de linha para controlar a ordem pela qual as linhas de programa são armazenadas em memória e executadas. Esta característica tem ainda a vantagem de tornar a edição de linhas de programa particularmente clara, devido ao facto de cada linha ter o seu número próprio como única referência.

Contudo, esta característica apresenta também algumas desvantagens de importância vária. Em primeiro lugar, constitui um desperdício de espaço, pois, na sua maioria, as linhas apenas são utilizadas durante o desenvolvimento do programa, e não durante a sua execução. Em segundo lugar, o programador tem, muitas vezes, de perder tempo a mudar de posição linhas de programa de forma a criar espaço suficiente para inserir uma nova rotina. Uma terceira desvantagem reside no facto de os números de linha serem também utilizados para identificar os destinos das instruções GOTO e GOSUB. Por este motivo, deve ter-se grande cuidado ao alterar números de linha, para que haja a certeza de que se alteram também todos os endereços de linha importantes incluídos em declarações GOTO e GOSUB.

Escrevemos esta rotina renumeradora de linhas com o fim de facilitar um pouco a tarefa de mudar os números de linha, embora o programador tenha ainda, ele próprio, de renumerar os números de linha incluídos nas declarações GOTO e GOSUB. O princípio pelo qual esta rotina funciona é muito simples e resume-se no pequeno programa seguinte, 11, o qual foi concebido de forma a renumerar-se a si próprio, começando em oito e utilizando um incremento de quatro. A rotina funciona indo buscar o endereço do primeiro *byte* da área de programas da RAM (e, por conseguinte, o endereço do primeiro de dois *bytes* que armazenam o primeiro número de linha), endereço esse obtido da variável de sistema PROG, contida nas células 23635 e 23636

(ver páginas 173 a 176 do manual de programação BASIC do *ZX Spectrum*). O novo número de linha é introduzido através de POKE no seu devido lugar pelas linhas 20 e 25. A linha 35 faz uso do comprimento da linha de programa, que está contido nas duas células imediatamente a seguir às que armazenam o número de linha, de forma a calcular o endereço da linha de programa seguinte. Desde que o novo endereço não fique situado para além do fim da área de programas (e, portanto, maior do que o valor da variável de sistema VARS situada nas células 23627 e 23628) o programa volta à linha 20 para renumerar a linha de programa seguinte.

Programa 11. *Um programa que se renumera a si próprio.*

```
  5 LET num=8
 10 LET ss=4
 15 LET som=PEEK 23635+256*PEE
23636
 20 POKE som,num/256
 25 POKE som+1,num-256*INT (nu
/256)
 30 LET num=num+ss
 35 LET som=som+4+PEEK (som+2)
256*PEEK (som+3)
 40 IF som<PEEK 23627+256*PEEK
23638 THEN GO TO 20
```

Este programa é uma simples sugestão exemplificativa, pois apenas funciona se a linha número 20 mantiver esse mesmo número durante a execução; de contrário o GOTO 20 da linha 40 não indicaria a linha correcta. Por aqui se vê que uma linguagem de programação superior ao BASIC trataria, muito provavelmente, os números de linha e os números de linha de referência *(labels)* incluídos nas declarações GOTO e GOSUB como entidades diferentes (as melhores linguagens evitam simplesmente o uso de declarações GOTO, neutralizando assim o problema).

A rotina completa possui ainda o requinte de poder renumerar uma parte seleccionada de um programa. Ela ocupa uma série de

linhas numeradas a partir de 9000, de forma a que, excepto no caso de se empregarem números de linha superiores a 8999, se possa facilmente fundir (MERGE) a um programa já existente.

Chama-se a rotina renumeradora introduzindo RUN 9000, pedindo ela logo a introdução dos números da primeira e da última linha a renumerar. Só aceita respostas coerentes e assim, por exemplo, números de linha negativos e números de linha superiores a 8999 são rejeitados. O programa (rotina) pesquisa a memória para encontrar o número de linhas que devem ser renumeradas e calcula o intervalo máximo permissível (valor do passo ou incremento) entre os novos números de linha. Em seguida, pede que se seleccione um novo número de linha inicial e a dimensão do intervalo (passo) e, desde que estes valores se incluam dentro dos limites permitidos, efectua a renumeração.

Deve sublinhar-se que esta rotina não renumera os *labels* (números de linha de referência) incluídos nas declarações GOTO e GOSUB. Seria quase impossível escrever uma rotina em BASIC que o fizesse. Contudo, a sequência de operação de uma rotina adequada em código máquina poderia ser:

1) Armazenar os endereços de todos os GOSUBs e GOTOs do programa e os números de linha de referência neles contidos numa área de trabalho adequada na memória;

2) Renumerar cada linha, à vez, comparando o número de linha primitivo com os números de linha incluídos nas declarações GOTO e GOSUB que se encontram na mencionada área;

3) Sempre que se encontre uma identidade converter o novo número de linha para a forma de carácter e imprimi-lo sobre o número de linha de referência (que se encontra expresso na forma de uma cadeia de caracteres) incluído na declaração GOTO ou GOSUB correspondente;

4) Se existir uma diferença de comprimento entre o primitivo e o novo número de linha de referência, expandir ou contrair a área de programa (por exemplo, se o número primitivo fosse 95 e o novo 140, a área de programa teria então de ser aumentada de um *byte,* pois 95 é escrito em dois caracteres, enquanto que 140 utiliza três). Seguidamente, actualizar os endereços contidos na área de trabalho mencionada, de acordo com o exposto;

5) Converter o novo número de linha para a forma de vírgula flutuante e escrevê-lo por cima da forma numérica do número de linha contido na declaração GOTO ou GOSUB, o qual se encontra oculto nos cinco *bytes* (mais um *byte* de separador) que se seguem à forma de carácter do dito número de linha de referência;

6) Apagar o lançamento feito na área de trabalho de forma a que os números de linha de referência não sejam acidentalmente renumerados mais do que uma vez.

Programa 11-A. «*Renumerador de linhas*».

```
9000 GO SUB 9970
9010 PRINT "Introduza o primeiro
 e o ultimo numero de linha da s
eccao a ser renumerada"
9020 INPUT a,b
9030 IF a<0 OR b<=a OR b>8999 OR
 a<>INT a OR b<>INT b THEN BEEP
.2,24: GO TO 9020
9040 GO SUB 9800
9050 GO SUB 9970
9060 PRINT "Actual numero inicia
l = ";num
9070 LET snum=onum: LET ssom=som
9080 LET a=b: LET m=n
9090 GO SUB 9800
9100 PRINT "Actual numero final
  = ";onum
9110 PRINT ,,"O novo num. ini. d
eve ser > ";snum
9120 PRINT "O novo num. fin. dev
e ser < ";num
9130 PRINT ,,"Num. de linhas a r
enumerar = ";n-m
9140 LET ss=INT ((onum-snum)/(n-
m))
9150 PRINT "O valor do passo dev
e ser <= ";ss
9200 PRINT "Introduza novo numer
o inicial e valor de passo:"
9210 INPUT a,b
9220 IF a<=snum OR b<1 OR b>ss O
R a<>INT a OR b<>INT b THEN BEEP
 .2,24: GO TO 9210
9300 PRINT ,,"Numero inicial = "
;TAB 27;a
9310 PRINT "Valor do passo = ";T
AB 27;b
9320 FOR i=1 TO n-m
9330 POKE ssom,INT (a/256)
9340 POKE ssom+1,a-256*INT (a/25
6)
9350 LET ssom=ssom+4+PEEK (ssom+
2)+256*PEEK (ssom+3)
9360 LET a=a+b
9370 NEXT i
9380 PRINT TAB 8; FLASH 1;"Renum
eracao terminada"
9390 STOP
9800 LET onum=0: LET osom=0: LET
 n=0
9810 LET som=PEEK 23635+256*PEEK
 23636
9820 LET num=256*PEEK som+PEEK (
som+1)
9830 IF num=a AND a<>b THEN RETU
RN
9840 IF num>a THEN RETURN
9850 LET onum=num: LET osom=som
9860 LET som=som+4+PEEK (som+2)+
256*PEEK (som+3)
9870 LET n=n+1
9880 GO TO 9820
9970 CLS
9980 PRINT TAB 6;"Renumerador de
 Linhas"'''
9990 RETURN
```

CARREGADOR DE CÓDIGO MÁQUINA

Este programa é muito útil para carregar em memória rotinas curtas em linguagem ou código máquina do género das que aparecem de tempos a tempos em publicações periódicas de informática, e incluímos um par de rotinas apropriadas como material de teste. A primeira rotina renumera um programa (exceptuando os números de linha de referência incluídos nos GOTOs e GOSUBs) e a segunda memoriza a imagem do *écran* e «chama-a» novamente ao *écran*. O «Carregador de código máquina» não se presta a ser desenvolvido, em contraste com outras rotinas de carregamento de código máquina, pois não

possui disponibilidades sofisticadas. Um programa mais adequado a tal trabalho é o «Editor de código máquina», também incluído neste volume.

O *menu* que controla o programa é impresso pela linha 500 e apresentado na figura 17. Selecciona-se uma opção do *menu,* fazendo com que o programa salte para a rotina adequada, por meio da linha 520.

```
                    Menu

Teclado para memoria                T
REM para memoria                    R
Apresentar hex                      a
Hex para REM                        i
Parar                               p
```

Figura 17. *O* menu *de «Carregador de código máquina».*

Tratamento dos endereços de RETURN

A opção «Teclado para memória» permite introduzir código máquina hexadecimal por meio do teclado. O programa solicita primeiramente, através da sub-rotina com início na linha 6000, o endereço no qual deve ser armazenada a forma absoluta (e não hexadecimal) do código. Em geral responder-se-ia introduzindo um endereço situado acima de RAMTOP.

Se o endereço se situar abaixo de RAMTOP, o programa pede a confirmação de que deve prosseguir. Se a resposta for «S» (de Sim), o programa converte o código hexadecimal introduzido subsequentemente para forma numérica e coloca-o, através de POKE, em memória, com início no endereço que foi dado.

Vale a pena reparar no que acontece se a resposta for «N» (de Não). O programa está agora numa sub-rotina mas não pode, no caso mencionado, executar o RETURN porque não desejamos prosseguir. Se executasse GOTO 600 imediatamente, deixaria o endereço do RETURN, ou seja, o número de linha para a qual deveria ter sido feito o retorno (RETURN), na pilha de GOSUB. Esta situação ocorre devido ao facto de não existir no BASIC *ZX* qualquer possibilidade de uma sub-rotina retornar (RETURN) a um de dois pontos de reentrada, dependendo de uma decisão tomada dentro da sub-rotina.

O programa contorna este problema introduzindo por meio de POKE um novo número de linha de RETURN, 599, na pilha de GOSUB, e executando seguidamente um RETURN. O novo número de linha de RETURN (nr) é fixado em 599 devido ao facto de o *software* da ROM incrementar o número. Assim o programa continua a partir da linha 600.

Visualiza-se a forma da pilha de GOSUB com o ciclo das linhas 9000 a 9030. Este ciclo não faz parte do programa principal e pode ser omitido, se se desejar. Os valores na listagem do programa referem-se aos endereços de um *Spectrum* de 48 k. Para introduzir o programa num computador de 16 k deve alterar-se 65367 para 32599.

Para ver a pilha de GOSUB, verifica-se primeiramente se não existem quaisquer endereços de RETURN redundantes na pilha, o que se faz introduzindo RETURN várias vezes, através do teclado, até a mensagem

7 RETURN without GOSUB, 0:1

aparecer no *écran*. Seguidamente, introduzindo RUN 9000, imprimir-se-á a tabela seguinte:

65367	62
65366	?
65365	19
65364	3
65363	27
65362	118
65361	127
65360	88
•	•
•	•

O valor na célula 65366 varia com as tarefas que a máquina tenha efectuado desde que foi ligada e é apresentado na tabela acima como «?». É esta a forma que a pilha apresenta quando não existem números de linha de RETURN.

Introduza-se agora uma nova linha:

9600 GOSUB 9700

e chame-se essa linha introduzindo RUN 9600. Fazer isto irá colocar o endereço de RETURN 9600 na pilha de GOSUB. A tabela impressa na linha 9000 apresentará então o seguinte aspecto:

65367	62
65366	?
65365	2
65364	37
65363	128
65362	19
65361	3
65360	27
•	•
•	•

Foram inseridos três *bytes*. O primeiro *byte* contém «2», o que indica que se segue um número de dois *bytes*. O segundo e o terceiro *byte* contêm, respectivamente, «37» e «128» e, no seu conjunto, formam o número de linha de RETURN, pois 37*256+128=9600. É nestas células de memória que a rotina das linhas 9500 a 9540 introduz por meio de POKE o novo número de linha de RETURN requerido.

O resto do «menu»

As outras opções do programa são bastante claras. «REM para memória» carrega o código hexadecimal contido numa declaração REM que forma a primeira linha do programa em código absoluto numa célula que especificarmos. A listagem exemplificativa da linha 10 apresenta a rotina «Renumerador de linhas» contida na forma descrita.

A opção «Apresentar hex» lista os códigos hexadecimais de um conjunto de células de memória consecutivas que seleccionarmos. ''Hex para REM'' converte o código absoluto contido num conjunto de células para código hexadecimal e coloca-o, por meio de POKE, na declaração REM. Esta opção serve, portanto, para copiar e modificar rotinas em código máquina da ROM.

Renumerador de linhas

A tabela 4 lista uma rotina para renumerar um programa (excepto GOTOs e GOSUBs) em passos de 10 com início na linha 10.

Tabela 4. *Renumerador de linhas (excepto GOSUBs e GOTOs) em código máquina.*

```
01 0A 00          LD BC, 10
11 0A 00          LD DE, 10
2A 53 5C          LD HL, (23635)
72       NEXTLOC  LD (HL), D
23                INC HL
73                LD (HL), E
23                INC HL
D5                PUSH DE
5E                LD E, (HL)
23                INC HL
56                LD D, (HL)
23                INC HL
19                ADD HL, DE
ED 5B 4B 5C       LD DE, (23627)
E5                PUSH HL
A7                AND A
ED 52             SBC HL, DE
D1                POP DE
E1                POP HL
D0                RET NC
09                ADD HL, BC
EB                EX DE, HL
18 E7             JR, -25  NEXT LOC
```

Memorizar e recuperar o «écran»

A tabela 5 apresenta duas rotinas destinadas a memorizar em 25856 a totalidade da imagem apresentada no *écran*, incluindo o ficheiro de atributos, e a copiá-la de volta para 16384. Deve ter-se o cuidado de voltar a colocar a RAMTOP a 25855 (ou menos) antes de a rotina de memorização ser executada e de se

verificar se as próprias rotinas não se encontram armazenadas entre 25856 e 32767.

Tabela 5. *Memorizar e carregar o* écran, *incluindo o ficheiro de atributos em/de 25856.*

Copiar *écran* para 25856

21 00 40	LD HL, 16384
11 00 65	LD DE, 25856
01 00 1B	LD BC, 6912
ED BO	LDIR
C9	RET

Copiar 25856 para *écran*

21 00 65	LD HL, 25856
11 00 40	LD DE, 16384
01 00 1B	LD BC, 6912
ED BO	LDIR
C9	RET

Programa 12. *«Carregador de código máquina».*

```
  10 REM 010a00110a002a535c72237
323d55e2358231 9ed5b4b5ce5a7ed52d
1e1d009eb18e7
 500 CLS : PRINT ,,TAB 14;"Menu"
,,,,,"Teclado para memoria";TAB
31;"TREM para memoria";TAB 31;"R
APresentar hex";TAB 31;"AHex par
a REM";TAB 31;"HParar";TAB 31;"P
"
 510 INPUT z$: LET z$=CHR$ (CODE
 z$-32*(z$>"£"))
 520 GO TO 600+400*(z$="T")+900*
(z$="R")+1400*(z$="A")+1900*(z$=
"H")+2400*(z$="P")
 599 REM
 600 INPUT "Prima ENTER para con
tinuar";z$
 610 GO TO 500
1000 PRINT ,,TAB 2;"Hex do tecla
do para memoria"
1010 GO SUB 6000
1020 PRINT ,,"Introduza hex ",,,
,
```



```
1030 INPUT z$: IF z$="" THEN GO
TO 600
1040 LET l=LEN z$
1050 IF l<>2*INT (l/2) THEN GO T
O 1440
1055 LET y$=z$
1060 FOR i=1 TO l
1065 LET y$(i)=CHR$ (CODE z$(i)-
32*(z$(i)>"£"))
1070 LET z$(i)=CHR$ (CODE z$(i)-
32*(z$(i)>"£")+7*(z$(i)<"A"))
1080 IF z$(i)<"7" OR z$(i)>"F" T
HEN GO TO 1400
1090 NEXT i
1100 PRINT a,y$
1110 FOR i=1 TO l STEP 2
1120 POKE a,16*CODE z$(i)+CODE z
$(i+1)-935
1130 LET a=a+1
1140 NEXT i
1150 GO TO 1030
1400 BEEP .2,24
1410 IF i=LEN z$ THEN GO TO 1440
1420 PRINT ,,z$(1 TO i); FLASH 1
;"?";z$(i+1 TO LEN z$)
1430 GO TO 1030
1440 PRINT ,,z$; FLASH 1;"?"
1450 GO TO 1030
1500 PRINT ,,TAB 4;"Hex de REM p
ara memoria"
1510 GO SUB 6000
1520 INPUT "Int. num. de caracte
res hex ";n
1530 IF n<1 OR n<>2*INT (n/2) TH
EN BEEP .2,24: GO TO 1520
1540 LET l=5+PEEK 23635+256*PEEK
 23636
1550 FOR i=l TO l+n-1 STEP 2
1560 LET j=PEEK i: LET k=PEEK (i
+1)
1570 LET j=j-32*(j>96)+7*(j<65)-
55
1580 LET k=k-32*(k>96)+7*(k<65)-
55
1590 IF j<0 OR j>15 OR k<0 OR k>
15 OR a>65535 THEN GO TO 600
1600 POKE a,16*j+k
1610 LET a=a+1
1620 NEXT i
1630 GO TO 200
2000 PRINT ,,TAB 10;"Apresentar
Hex"
```

```
2010 INPUT "Intr. endereco inici
al p/ codigo";a: IF a<0 OR a>IN
T a OR a>65536 THEN BEEP .2,24:
GO TO 2010
2020 LET fa=a
2030 IF fa-a=8*INT ((fa-a)/8) TH
EN PRINT a;TAB 8;
2035 LET i=INT (PEEK a/16)
2040 PRINT CHR$ (i+55-7*(i<10));
2050 LET i=PEEK a-16*i
2060 PRINT CHR$ (i+55-7*(i<10));
" ";
2070 IF INKEY$="s" OR INKEY$="S"
 THEN GO TO 600
2080 LET a=a+1
2090 GO TO 2030
2200 INPUT "Intr. endereco inici
al p/ codigo";a: IF a<16384 OR a
<>INT a THEN BEEP .2,24: GO TO 6
000
2500 PRINT ,,TAB 11;"Hex p/ REM"
2510 LET p=PEEK 23635+256*PEEK 2
3636
2520 LET l=PEEK (p+2)+256*PEEK (
p+3)
2530 IF PEEK (p+4)<>234 THEN BEE
P .2,24: PRINT ,,"A primeira dec
laracao nao e' REM": GO TO 600
2540 INPUT "Intr. endereco inici
al p/ codigo";a: IF a<0 OR a>655
36 OR a<>INT a THEN BEEP .2,24:
GO TO 2540
2550 INPUT "Int. num. de bytes a
 copiar ";n: IF n<0 THEN BEEP .2
,24: GO TO 2550
2560 IF n>l/2 THEN BEEP .2,24: P
RINT ,,"So ha' espaco para ";INT
 l/2;" bytes": GO TO 600
2570 LET p=p+5
2580 FOR i=a TO a+n-1
2590 LET j=INT (PEEK i/16)
2600 POKE p,j+55-7*(j<10)
2610 LET j=PEEK i-16*j
2620 POKE p+1,j+55-7*(j<10)
2630 LET p=p+2
2640 NEXT i
2650 GO TO 600
3000 PRINT ,,"O programa parou"
3010 STOP
6000 INPUT "Intr. endereco inici
al p/ codigo";a: IF a<16384 OR a
<>INT a THEN BEEP .2,24: GO TO 6
```



```
000
6010 IF a>PEEK 23730+256*PEEK 23
731 THEN RETURN
6020 BEEP .2,24: INPUT "O endere
co inicial e' menor que RAMTOP.
Quer continuar (S ou N)?";z$
6030 IF z$="S" OR z$="s" THEN RE
TURN
6040 LET nr=599
6050 GO TO 9500
9000 FOR i=0 TO 20
9010 PRINT 65367-i,PEEK (65367-i
)
9020 NEXT i
9030 STOP
9500 LET r=PEEK 23730+256*PEEK 2
3731-2
9510 IF PEEK r=2 THEN LET r=r-3:
 GO TO 9510
9520 POKE r+1,nr-256*INT (nr/256
)
9530 POKE r+2,INT (nr/256)
9540 RETURN
```

DESASSEMBLADOR

O leitor sabe, provavelmente, que o microprocessador *Z80* que comanda o *ZX Spectrum* não compreende directamente palavras de BASIC, tais como PRINT, IF, TAB, etc. Em vez disso, obedece a uma linguagem chamada «código máquina» ou «linguagem máquina». As instruções na ROM *Sinclair,* por exemplo, estão escritas neste código. O código máquina consiste numa sequência de números inteiros positivos, cada um deles menor que 256. A função de um programa desassemblador é converter a sequência de números inteiros numa sequência de mnemónicas assembler mais facilmente compreensível (um programa assemblador tem a função complementar de converter mnemónicas assembler em números inteiros positivos).

Para ver alguns exemplos de mnemónicas deste tipo, consulte-se o apêndice A do manual de programação BASIC do *ZX Spectrum*, começando na página 183. A primeira coluna, com o cabeçalho «*Code*» (código), lista os inteiros positivos de 0 a 255 e a quarta coluna, com o cabeçalho «*Z80* Assembler», lista uma mnemónica para cada número inteiro. Por exemplo, a mnemónica «nop» é a abreviatura de "no operation" (não

operação) e refere-se ao código máquina 0, devido ao facto de 0 ser a instrução «não fazer nada». Por outras palavras, o *Z80* executa a instrução de código máquina «0» não fazendo nada até ser altura de executar a instrução seguinte.

Uma outra mnemónica simples é «inc b» (código máquina 4), a qual instrui o *Z80* para incrementar em 1 o registo b. Por outras palavras, se o registo b contiver 210 antes de a instrução ser executada, conterá 211 após a execução.

O conjunto de instruções do *Z80* é complexo e não cabe no âmbito deste livro descrever a sua forma e estrutura.

O programa desassemblador tem de interpretar correctamente todas as complexidades do conjunto de instruções do *Z80*. Existem cerca de 640 instruções diferentes neste processador. Algumas, como «nop» e «inc b», são completamente especificadas em um inteiro, mas outras, como «ld b, N» (código máquina 6) necessitam de dois, três ou mesmo quatro inteiros. A instrução «ld b, N» significa «carregar *(load)* o registo b com a variável N que se segue à instrução». Assim, o par de valores de código máquina 6, 15 faz com que o *Z80* carregue o registo b com o valor 15. As variáveis são indicadas por letras maiúsculas na coluna quatro do apêndice A do manual de programação BASIC do *ZX Spectrum*.

A outra forma pela qual as instruções necessitam de mais do que um inteiro de modo a serem completamente definidas é pela utilização de quatro números (203, 221, 237 e 253; CB, DD, ED e FD em hexadecimal) para formar um código ampliado. Os códigos ampliados CD e ED são listados nas colunas 5 e 6 do apêndice A. Assim, o par 237, 00 (CD 00 em hexadecimal) é o código correspondente a «rlc b», o qual significa «deslocar para a esquerda, com transporte, todo o registo b» *(rotate left with carry register b)*. Por outras palavras, cada um dos oito *bits* no registo b é deslocado de um *bit* para a esquerda, sendo o *bit* mais à esquerda copiado tanto para o «indicador» de transporte *(carry flag)* como para a posição anteriormente ocupada pelo *bit* mais à direita.

No apêndice A não se listam os códigos ampliados DD e FD

devido ao facto de serem idênticos aos outros códigos, embora com os registos de 16 *bits* ix e iy substituídos por hl. Por exemplo, o código máquina 35 («inc hl») instruiu o processador *Z80* de modo a incrementar o registo de 16 *bits* hl. Analogamente, o par de instruções de código máquina 221, 35 (DD, 23 em hexadecimal) significa «inc ix» — incrementar o registo ix; 253, 35 (FD, 23 em hexadecimal) significa «inc iy» — incrementar o registo iy.

Concepção do programa

A forma mais óbvia de escrever um programa desassemblador consiste em elaborar uma tabela de consulta contendo cada uma das 640 mnemónicas «oficiais» (existem ainda algumas combinações de códigos «não oficiais» que, não intencionalmente, também funcionam). Contudo, um cálculo rápido mostra que não existe espaço suficiente no *Spectrum* não ampliado (normal) para conter uma tal tabela juntamente com um programa adequado. Assumindo, por exemplo, que a mnemónica genérica consiste em seis caracteres e que, para cada mnemónica, é necessário um indicador *(pointer)* numérico que ocupe cinco *bytes*, a tabela e os indicadores ocupariam só por si um mínimo de $(6+5)\times640=7040$ *bytes*. Seria também possível utilizar indicadores inteiros (ocupando dois *bytes* cada), em vez da forma usual da Sinclair, de cinco *bytes*, o que reduziria o espaço necessário para $(6+2)\times640=5120$ *bytes*.

Uma abordagem do problema mais elegante do que esta consiste em tirar partido das simetrias de partes do conjunto de instruções para construir algumas das mnemónicas a partir das suas partes componentes. Os códigos máquina de um *byte* entre 64 e 191 inclusive, por exemplo, dizem todos respeito aos registos b, c, d, e, h, l, (hl) e a, numa sequência periódica. Os números 64 a 127 são todos instruções de carregamento («ld» — *load*) e 128 a 191 seguem-se-lhes em grupos de oito — add a; adc a; sub, etc. Analogamente, os códigos de dois *bytes* com início em 203 (CB em hexadecimal), tal como aparecem listados na coluna cinco do apêndice A, consistem em oito grupos de oito seguidos de três grupos de 64. Estes códigos também apresentam

a mesma sequência repetitiva de referências aos registos b, c, d, e, h, l, (hl) e a.

A tabela 6 mostra como o desassemblador analisa o código máquina. É utilizado um indicador, w$, para fazer a distinção entre os registos hl, ix e iy. Esse indicador é alterado adequadamente se ocorrer o código decimal 221 ou 253 (DD e FD em hexadecimal). Se ocorrer um código ampliado ED (237 em decimal), a mnemónica é procurada na cadeia de caracteres y$. Se, em vez disso, ocorrer um código ampliado CB (203 em decimal), a mnemónica inteira é construída a partir das suas partes componentes. Uma instrução de um *byte* tanto pode ser lida da cadeia de caracteres z$ como construída a partir das suas partes componentes.

O desassemblador em pormenor

O programa consiste em duas secções distintas. A primeira secção (linhas 10 a 1340) carrega em memória as tabelas de consulta z$ e y$, além dos seus indicadores p e q. A segunda secção (linhas 3000 a 9010) é o próprio desassemblador. Na sua forma completa, o programa ocupa a quase totalidade da memória de um *Spectrum* de 16 k e, para o introduzir na totalidade, é necessário colocar a RAMTOP no seu valor mais elevado, introduzindo CLEAR 32767.

Tabela 6. *O processo utilizado pelo «Desassemblador» para analisar os vários códigos máquina.*

Código Máquina (Decimal)	(Hex)	Procedimento
0 a 63	00 a 3F	Primeiros 64 elementos de z$
64 a 191	40 a BF	Construídos nas linhas 3400 a 3510
192 a 255	CO a FF	Últimos 64 elementos de z$
203,0 a 255	CB,00 a FF	Construídos nas linhas 4500 a 4810
237, 64 a 123	ED,40 a 7B	Primeiros 60 elementos de y$
237,160 a 187	ED,AO a BB	Últimos 28 elementos de y$
221,9 a 249	DD,09 a F9	Por ajustamento de y$ nas linhas 4100 a 4150
253,9 a 249	FD,09 a F9	Por ajustamento de z$ nas linhas 4100 a 4150
221,203,6 a 254	DD,C9,06 a FE	Construídos nas linhas 3400 a 3510 e ajustados pela sub-rotina em 8000
253,203,6 a 254	FD,C9,06 a FE	Construídos nas linhas 3400 a 3510 e ajustados pela sub-rotina em 8000

De facto, uma vez carregados os dados por execução (RUN) do programa, as linhas 10 a 1340 podem ser apagadas na sua totalidade *desde que as instruções RUN e CLEAR nunca sejam utilizadas subsequentemente*. A melhor forma de acção consiste em executar todo o programa, gravar uma cópia, apagar as linhas 10 a 1340 e gravar uma segunda cópia introduzindo

SAVE «Desassemb» LINE 3000

Deste modo, a cópia original completa fica disponível como apoio. Sempre que carregamos a segunda cópia em memória, a sua execução é iniciada na linha 3000 sem que as variáveis sejam apagadas.

As mnemónicas variam em comprimento e, por esse motivo, cada uma das duas tabelas z$ e y$ necessita de um quadro indicador *(pointer array)* p e q para indicar, à vez, o fim de cada mnemónica. A primeira secção do programa carrega nas duas tabelas os dados contidos nas declarações DATA. É claro que não tem qualquer importância se são utilizadas onze declarações DATA, como se fez, se mais, se menos, desde que se mantenha a sequência correcta.

A técnica utilizada na segunda secção do programa, nas linhas 3400 a 3510 por exemplo, para lidar com as declarações DATA, é digna de nota. É esta a parte do programa que constrói instruções da forma ld e,b, e fá-lo «lendo» o elemento adequado na declaração DATA da linha 3400, armazenando o resultado e, seguidamente, «lendo» a mesma declaração uma segunda vez. O número de elementos lidos é determinado pelos «contadores» de ciclo nas linhas 3440 e 3480, os quais são calculados a partir do conteúdo do *byte* a analisar.

A tabela 7 é um exemplo de *output* do programa. Apresenta-se à vez cada endereço e mnemónica.

Tabela 7. *Um exemplo de* output *do programa «Desassemblador».*

```
1500    or  e
1501    jr  nz,-54
1503    ld  a,h
1504    cp  1
1506    ret
1507    call  1511
1510    ret  nc
1511    ld  a,22
1513    dec  a
1514    jr  nz,-3
1516    and  a
1517    inc  b
1518    ret  z
1519    ld  a,127
1521    in  a,(254)
1523    rra
1524    ret  nc
1525    xor  c
1526    and  32
1528    jr  z,-13
1530    ld  a,c
1531    cpl
```

Programa 13. *«Desassemblador».*

```
  5 REM "DESASSEMBLADOR"
 10 REM Esta seccao carrega em
memoria os codigos de op. das
colunas 4 e 5 do apendice A do
manual de programacao Basic do
ZX Spectrum
 100 DIM p(128)
 110 DIM z$(809)
 120 DIM q(88)
 130 DIM y$(392)
 500 DATA "ld  bc,NN","ld  (bc),a"
,"inc  bc","inc  b","dec  b","ld  b,
N","rlca","ex  af,af'","add  hl,bc
","ld  a,(bc)","dec  bc"
 501 DATA "inc  c","dec  c","ld  c,
N","rrca","djnz  DIS","ld  de,NN","
ld  (de),a","inc  de","inc  d","de
c  d","ld  d,N","rla","jr  DIS","add
 hl,de","ld  a,(de)","dec  de","ini
nc  e","dec  e","ld  e,N","rra","jr
 nz,DIS","ld  hl,NN","ld  (NN),hl"
,"inc  hl","inc  h","dec  h"
```



```
 502 DATA "ld  h,N","daa","jr  z,D
IS","add  hl,hl","ld  hl,(NN)","de
c  hl","inc  l","dec  l","ld  l,N","
cpl","jr  nc,DIS","ld  sp,NN","ld(
NN),a","inc  sp","inc  (hl)","dec
(hl)","ld  (hl),N","scf","jr  c,DI
S","add  hl,sp","ld  a,(NN)"
 503 DATA "dec  sp","inc  a","dec
a","ld  a,N","ccf"
 510 DATA "ret  nz","pop  bc","jp
nz,NN","jp  NN","call  nz,NN","pus
h  bc","add  a,N","rst  0","ret  z",
"ret","jp  z,NN","","call  z,NN","
call  NN","adc  a,NN","rst  8","ret
 nc","pop  de","jp  nc,NN","out  (N
),a","call  nc,NN","push  de"
 511 DATA "sub N","rst  16","ret
c","exx","jp  c,NN","in  a,(N)","c
all  c,NN","","sbc  a,N","rst  24",
"ret  po","pop  hl","jp  po,NN","ex
 (sp),hl","call  po,NN","push  hl"
,"and  N","rst  32","ret  pe","jp
hl)"
 512 DATA "jp  pe,NN","ex  de,hl",
"call  pe,NN","","xor  N","rst  40"
,"ret  p","pop  af","jp  p,NN","di"
,"call  p,NN","push  af","or  N","r
st  48","ret  m","ld  sp,hl","jp  m,"
NN","ei","call  m,NN","","cp  N","
rst  56"
 520 DATA "out  (c),b","sbc  hl,bc
","ld  (NN),bc","neg","retn","im
0","ld  i,a","in  c,(c)","out  (c),
c","adc  hl,bc","ld  bc,(NN)","","
reti","","ld  r,a","in  d,(c)","ou
t  (c),d","sbc  hl,de","ld  (NN),de
"
 521 DATA "","","im  1","ld  a,i",
"in  e,(c)","out  (c),e","adc  hl,d
e","ld  de,(NN)","","","im  2","ld
 a,r","in  h,(c)","out  (c),h","sb
c  hl,hl","ld  (NN),hl","","","","
rrd","in  l,(c)","out  (c),l","adc
 hl,hl","ld  hl,(NN)"
 522 DATA "","","","rld","in  f,(
c)","","sbc  hl,sp","ld  (NN),sp",
"","","","","in  a,(c)","out  (c),
a","adc  hl,sp","ld  sp,(NN)"
 530 DATA "ldi","cpi","ini","out
i","","","","ldd","cpd","ind"
,"outd","","","","","ldir","cpir
","inir","otir","","","","","ldd
```

```
r","cpdr","indr","otdr"
1000 LET p(1)=3
1010 LET z$(1 TO 3)="nop"
1020 FOR i=1 TO 63
1030 READ i$: PRINT i;".......";
i$
1040 LET p(i+1)=p(i)+LEN i$
1050 LET z$(p(i)+1 TO p(i+1))=i$
1060 NEXT i
1100 FOR i=192 TO 255
1110 READ i$: PRINT i;".......";
i$
1120 LET p(i-127)=p(i-128)+LEN i
$
1130 LET z$(p(i-128)+1 TO p(i-12
7))=i$
1140 NEXT i
1200 LET q(1)=8
1210 LET y$(1 TO 8)="in b,(c)"
1220 FOR i=65 TO 123
1230 READ i$: PRINT "ED  ";i;"..
.....";i$
1240 LET q(i-63)=q(i-64)+LEN i$
1250 LET y$(q(i-64)+1 TO q(i-63)
)=i$
1260 NEXT i
1300 FOR i=160 TO 187
1310 READ i$: PRINT "ED  ";i;"..
.....";i$
1320 LET q(i-99)=q(i-100)+LEN i$
1330 LET y$(q(i-100)+1 TO q(i-99
))=i$
1340 NEXT i
2990 REM ************************
************************
2992 REM A seccao seguinte e' o
         desassemblador
3000 CLS
3010 PRINT TAB 9;"Desassemblador
"
3020 PRINT ,,"Intr. os enderecos
 em que quer  comecar e terminar
."
3030 INPUT sa,sf
3040 IF sa<0 OR sf<sa OR sa<>INT
 sa OR sf<>INT sf THEN BEEP .2,2
4: GO TO 3030
3050 PRINT ,,"Deseja copia impre
ssa (S ou N) ?"
3060 INPUT c$: LET c$=CHR$ (CODE
 c$-32*(c$>"z")): IF c$<>"S" AND
 c$<>"N" THEN BEEP .2,24: GO TO
3060
```



```
3100 LET i=sa
3110 LET j=PEEK i
3120 PRINT i;TAB 7;: IF c$="S" T
HEN LPRINT i;TAB 7;
3130 LET w$="HL"
3140 GO TO 3200+1300*(j=203)+180
0*(j=237)+2300*(j=221)+2500*(j=2
53)
3200 IF j<>0 THEN GO TO 3230
3210 PRINT "nop": IF c$="S" THEN
 LPRINT "nop"
3220 GO TO 9000
3230 IF j>63 AND j<192 THEN GO T
O 3400
3235 LET t=j-128*(j>191)
3240 FOR k=p(t)+1 TO p(t+1)
3250 IF z$(k TO k+1)="NN" THEN G
O TO 3800
3260 IF z$(k)="N" THEN GO TO 390
0
3270 IF z$(k TO k+2)="DIS" THEN
GO TO 4000
3280 IF z$(k TO k+1)="hl" AND w$
<>"HL" THEN GO TO 4100
3300 PRINT z$(k);: IF c$="S" THE
N LPRINT z$(k);
3310 NEXT k
3320 PRINT : IF c$="S" THEN LPRI
NT
3330 GO TO 9000
3400 DATA "b","c","d","e","h","l
","(hl)","a"
3405 IF j<>118 THEN GO TO 3410
3406 PRINT "halt": IF c$="S" THE
N LPRINT "halt"
3407 GO TO 9000
3410 IF j>127 THEN GO TO 3530
3420 LET i$="ld"
3430 RESTORE 3400
3440 FOR k=1 TO j/8-7
3450 READ j$
3460 NEXT k
3470 RESTORE 3400
3480 FOR k=1 TO j+1-8*INT (j/8)
3490 READ k$
3500 NEXT k
3510 PRINT i$;" ";j$;",";k$: IF
c$="S" THEN LPRINT i$;" ";j$;","
;k$
3520 GO TO 9000
3530 DATA "add a,","adc a,","sub
 ","sbc a,","and ","xor ","or ",
"cp "
```

```
3540 RESTORE 3530
3550 FOR k=1 TO j/8-15
3560 READ i$
3670 NEXT k
3680 REM
3690 RESTORE 3400
3700 FOR k=1 TO j+1-8*INT (j/8)
3710 READ j$
3720 NEXT k
3730 PRINT i$;j$: IF c$="S" THEN
 LPRINT i$;j$
3740 GO TO 9000
3800 LET t=PEEK (i+1)+256*PEEK (
i+2)
3810 PRINT t;: IF c$="S" THEN LP
RINT t;
3820 LET i=i+2
3830 LET k=k+1
3840 GO TO 3310
3900 LET t=PEEK (i+1)
3910 PRINT t;: IF c$="S" THEN LP
RINT t;
3920 LET i=i+1
3930 GO TO 3310
4000 LET t=PEEK (i+1)-256*(PEEK
(i+1)>127)
4010 PRINT t;: IF c$="S" THEN LP
RINT t;
4020 LET i=i+1
4030 LET k=k+3
4040 GO TO 3310
4100 PRINT w$;: IF c$="S" THEN L
PRINT w$;
4110 LET k=k+1
4120 IF z$(k+1)<>")" OR j=233 TH
EN GO TO 3310
4130 LET i=i+1
4140 PRINT "+";PEEK i;: IF c$="S
" THEN LPRINT "+";PEEK i;
4150 GO TO 3310
4500 LET i=i+1+(w$<>"HL")
4510 LET j=PEEK i
4520 IF j>63 THEN GO TO 4700
4530 IF j<48 OR j>55 THEN GO TO
4560
4540 PRINT "Nao ha' codigo de op
.": IF c$="S" THEN LPRINT "Nao h
a' codigo de op."
4550 GO TO 9000
4560 DATA "rlc","rrc","rl","rr",
"sla","sra","","srl"
4570 RESTORE 4560
```



```
4580 FOR k=0 TO j/8
4590 READ i$
4600 NEXT k
4610 RESTORE 3400
4620 FOR k=0 TO j-8*INT (j/8)
4630 READ j$
4640 NEXT k
4645 IF j$="(hl)" AND w$<>"HL" T
HEN GO SUB 8000
4650 PRINT i$;" ";j$: IF c$="S"
THEN LPRINT i$;" ";j$
4660 GO TO 9000
4700 DATA "bit","res","set"
4710 RESTORE 4700
4720 FOR k=1 TO j/64
4730 READ i$
4740 NEXT k
4750 RESTORE 3400
4760 FOR k=0 TO j-8*INT (j/8)
4770 READ j$
4780 IF j$="(hl)" AND w$<>"HL" T
HEN GO SUB 8000
4790 NEXT k
4800 LET t=INT (j/8)-8*INT (j/64
)
4810 PRINT i$;" ";t;",";j$: IF c
$="S" THEN LPRINT i$;" ";t;",";j
$
4820 GO TO 9000
5000 LET i=i+1
5010 LET j=PEEK i
5020 IF j>63 AND j<124 THEN GO T
O 5100
5030 IF j>159 AND j<187 THEN GO
TO 5300
5040 GO TO 4540
5100 LET j=j-64
5105 IF j=0 THEN GO TO 5400
5110 IF q(j)+1=q(j+1) THEN GO TO
 4540
5120 FOR k=q(j)+1 TO q(j+1)
5130 IF y$(k TO k+1)="NN" THEN G
O TO 5200
5150 PRINT y$(k);: IF c$="S" THE
N LPRINT y$(k);
5160 NEXT k
5165 PRINT : IF c$="S" THEN LPRI
NT
5170 GO TO 9000
5200 LET t=PEEK (i+1)+256*PEEK (
i+2)
5210 PRINT t;: IF c$="S" THEN LP
```

```
RINT t;
5020 LET i=i+2
5030 LET k=k+1
5040 GO TO 5150
5300 LET j=j-100
5310 IF q(j)+1=q(j+1) THEN GO TO
 +540
5320 PRINT y$(q(j)+1 TO q(j+1));
 IF c$="S" THEN LPRINT y$(q(j)+1
 TO q(j+1))
5330 GO TO 9000
5400 PRINT y$(1 TO 8); IF c$="S"
 THEN LPRINT y$(1 TO 8)
5410 GO TO 9000
5500 LET w$="ix"
5510 LET i=i+1
5520 LET j=PEEK i
5530 GO TO 3140
5700 LET w$="iy"
5710 GO TO 5510
8000 IF PEEK (i-1)=253 OR PEEK (
i-1)=221 THEN GO TO 8030
8010 LET )$="(ix"+w$+"+"+STR$ PEEK
 (i-1)+")"
8020 RETURN
9000 LET i=i+1
9010 IF i<=s; THEN GO TO 3110
```

EDITOR DE CÓDIGO MÁQUINA

Com qualquer linguagem de programação, em qualquer computador, parece haver sempre tarefas que queiramos que a máquina execute mas que não podem ser programadas convenientemente na linguagem disponível. O *ZX Spectrum* não constitui excepção neste aspecto.

Considere-se, por exemplo, o problema de gravar a totalidade da imagem do *écran* no topo da RAM. Em conjunto, o ficheiro de imagem e o de atributos ocupam 6912 *bytes* e, por isso, é necessário deslocar a RAMTOP para baixo, para o endereço 32768−6912=25856 no computador de 16 k ou para o endereço 65536−6912=58624 no de 48 k. O pequeno programa seguinte,

em BASIC, efectua a gravação da imagem do *écran* no topo da RAM, mas leva muito tempo a fazê-lo — cerca de 70 segundos:

```
10 FOR i=0 TO 6911
20 POKE 25856+i, PEEK (16384+i)
30 NEXT i
```

A razão do excessivo tempo de execução do programa reside no facto de o *Spectrum* perder a maior parte do tempo a descodificar comandos, converter números entre a forma de inteiro de dois *bytes,* que o *Z80* compreende, e a forma decimal de cinco *bytes,* na qual o contador de ciclo é representado, e ainda a efectuar aritmética de cinco *bytes*. Os passos são os seguintes:

1) Adicionar i a 16384
2) Converter o resultado para a forma de dois *bytes*
3) Recuperar o conteúdo do endereço PEEK
4) Adicionar i a 25856
5) Converter o resultado para a forma de dois *bytes*
6) Armazenar o valor recuperado no endereço POKE
7) Adicionar 1 ao valor de i e armazenar o resultado
8) Subtrair i de 6911. Se o resultado for positivo ou zero, regressar a 1)

De cada vez que executa o ciclo, o *Spectrum* tem de descodificar novamente cada comando porque não memoriza as operações anteriores. É fácil de ver que o computador consome mais de 99 % do tempo a preparar-se para executar a tarefa em vez de o consumir propriamente a executá-la. Deste modo, não constitui surpresa saber que uma rotina em código máquina destinada a gravar a imagem do *écran* na RAM executa a tarefa quase instantaneamente. Damos uma rotina exemplificativa com o programa «Carregador de código máquina».

Infelizmente, escrever código máquina é uma tarefa difícil, e não é aconselhável tentar fazê-lo sem o auxílio de algum *software*. O melhor tipo de programa para escrever código máquina (em oposição a uma linguagem compilada) é, sem dúvida, um assemblador, ou assembler, em inglês. Contudo, um assemblador tem, igualmente, de ser escrito em código máquina

e não é praticável publicar a listagem de um longo programa em código máquina. Por esse motivo, incluiu-se no presente livro este «Editor de código máquina» em vez de um assemblador.

O «Editor» apresenta todas as funções importantes de um assemblador, com a excepção da possibilidade de conversão das mnemónicas assembler em hexadecimal. Utilizando o «Editor», é possível escrever código máquina sob a forma de caracteres em hexadecimal numa área de memória reservada, inserir e apagar partes do código, localizar e nomear variáveis de 8 e de 16 *bits*, nomear segmentos do código, chamar ou «saltar» para esses segmentos por meio dos respectivos nomes e carregar o código absoluto (código máquina «puro») no extremo superior da RAM, a partir da sua forma de carácter. O «Editor» é inteiramente escrito em BASIC e, por isso, as funções de inserção, apagamento e carregamento são bastante lentas — levando uma média de 5 a 20 segundos a executar.

Descrição sumária do programa

O programa é comandado a partir do *menu* impresso pela linha 2470 e apresentado na figura 18. A linha 2490 chama várias sub-rotinas, dependendo da opção seleccionada no *menu*.

```
Editor de Codigo Maquina

          Menu

Acrescentar uma REM          A
Carregar uma rotina          C
Nomear um segmento           N
Descarregar uma rotina       D
Inserir hex                  I
Apagar hex                   X
Rever hex                    R
Escrever hex                 E
Parar                        P

Espaco hex de 23760 a 23882
```

Figura 18. Menu *do «Editor de código máquina»*.

Acrescentar uma REM Introduzir A

O código hexadecimal introduzido através do teclado é armazenado na linha 1. O programa mantém um registo do número de posições disponíveis na declaração REM e apresenta o resultado logo abaixo do menu. Se for necessário mais espaço, temos de parar o programa, introduzir uma declaração REM de comprimento adequado numa linha 2, fazer executar o programa e, seguidamente, seleccionar a opção respectiva. A nova REM fica incorporada na anterior e a quantidade de espaço disponível para código hexadecimal aumenta em concordância.

A rotina que domina esta opção situa-se nas linhas 2430 a 2460 e 2610 a 2660. Quando se executa o programa, o comprimento da primeira linha de programa é armazenado em ℓ e efectua-se uma verificação para ver se a segunda declaração é uma REM (isto é, que o quinto *byte* na área de programa contém 234). Quando se selecciona a opção «Acrescentar uma REM», faz-se uma verificação para ver se a segunda declaração é uma REM e calcula-se um novo valor de ℓ, introduzindo por meio de POKE no indicador de comprimento de linha da primeira linha.

Escrever hex Introduzir E

A rotina que domina esta opção situa-se nas linhas 2680 a 2850. Temos de introduzir o endereço, situado dentro do espaço hex indicado a partir do qual desejamos que o código seja armazenado e somos, seguidamente, convidados a inserir o código em grupos adequados. O programa rotula automaticamente o fim de cada grupo, acrescentando 128 ao código de carácter, de forma a que a opção «Rever hex» possa separar o código nos mesmos grupos. O programa copia para o *écran* o que introduzimos pelo teclado juntamente com os endereços nos quais os códigos são armazenados. Devemos introduzir sempre H (de *halt* — parar) como último código de um programa, pois a rotina de carregamento pára quando encontra um H. Para terminar uma introdução no meio de um programa, utilizar T.

O exemplo seguinte mostra como se pode escrever no espaço hex uma rotina para imprimir o indicador de pilha.

Introdução	Comentário
E	Seleccionar a opção «Escrever hex»
23760	Endereço do início do espaço hex
210000	LD HL,0
39	ADD HL, SP
44	LD B, H
4D	LD C, L
C9	RET
H	Sinal de paragem para o carregador
T	Terminar introdução

Rever hex Introduzir R

Esta rotina encontra-se nas linhas 2900 a 3120. O endereço de cada grupo de códigos é impresso seguido de pares de códigos, à vez.

Inserir hex e apagar hex Introduzir I ou D

Podemos inserir ou apagar código hexadecimal em qualquer ponto do espaço hex. Em ambos os casos, deve ter-se o cuidado de introduzir o número de *caracteres* a inserir ou apagar e *não* o número de *bytes* do código compilado. Por exemplo, o código hexadecimal para «inc hl» é 23, isto é, dois caracteres, apesar de o código compilado ocupar apenas um *byte*.

A rotina de apagamento situa-se nas linhas 2340 a 2400 e a de inserção nas linhas 2160 a 2220. Depois de ter criado o número de espaços necessários, a rotina de inserção «salta» para a rotina de escrita («Escrever hex») de forma a que possamos criar espaço para escrever o código subsequente, dentro da mesma opção.

Nomear um segmento Introduzir N

Esta rotina, nas linhas 3130 a 3300, insere um nome numa célula escolhida por nós. O nome é precedido e seguido por um ponto final quando se insere na REM, tornando-se fácil de identificar tanto por nós como pelo programa. Os dois *bytes* que se seguem ao nome são utilizados para armazenar um ponto de entrada para o segmento, relativo à posição do nome. Este «dispositivo» permite-nos colocar uma sub-rotina «local» no início de um

segmento e passar por cima dela quando o segmento é chamado pela primeira vez.

Chamar segmentos e nomear variáveis

A opção de escrita pressupõe código hexadecimal como entrada mas, se introduzirmos o nome de um segmento, precedido e seguido por um hífen ou traço de união, esse nome também é escrito na declaração REM. Também podemos definir variáveis de 8 e de 16 *bits*, de modo semelhante, sendo essas variáveis aceites na forma S1, S2, S3, etc. (variáveis de 8 *bits*) e L1, L2, L3, etc. (variáveis de 16 *bits*).

Carregar uma rotina Introduzir C

Esta é a rotina mais complicada do programa. Primeiramente, ela pede-nos o endereço no qual desejamos que a versão absoluta do código seja armazenada. A resposta usual seria um endereço da área acima de RAMTOP. Se respondermos com um endereço que se situe abaixo do apontado pelo indicador STKEND, isto é, na área ocupada pelo próprio «Editor de código máquina», o programa pede uma confirmação do endereço.

A rotina efectua duas passagens pelo código hexadecimal contido na declaração REM. Na primeira passagem, efectua a contagem do número de diferentes variáveis curtas de 8 *bits* e do número de variáveis longas de 16 *bits* e armazena os nomes de todos os segmentos existentes no programa em y$. Antes de iniciar a segunda passagem, a rotina reserva espaço suficiente para as variáveis no início da área de código absoluto e calcula os pontos de entrada absolutos de todos os segmentos.

Na segunda passagem, a rotina reúne o código no lugar, substituindo os endereços absolutos das variáveis e dos segmentos de forma adequada.

Descarregar uma rotina Introduzir D

Esta rotina pode ser utilizada para verificar o código absoluto. Imprime o código hexadecimal para cada *byte* entre os limites que especificarmos.

Um exemplo

Para obter uma imagem intrigante, introduza-se o seguinte:

Introdução	Comentário
9500 POKE 31000,0	Introduzir, por meio de POKE,
9510 POKE 31001,64	o endereço do início
9520 POKE 31002,255	da imagem e do fim dos
9530 POKE 31003,91	atributos em 31000-31003
9540 RANDOMIZE	
USR 31004	Chamar a rotina de código
9550 GOTO 9540	máquina
RUN	Fazer executar o programa
E	Escrever uma rotina
23760	com início em 23760
2AL1	LD HL,(L1)
EB	EX,DE,HL
2AL2	LD HL,(L2)
A7	AND A
ED 52	SBC HL,DE
44	LD B,H
4D	LD C,L
2AL1	LD HL, (L1)
3A785C	LD A,(23672)
77	LD(HL),A
23	INC HL
0B	DEC BC
78	LD A,B
BC	OR C
FE00	CP,0
C2-NEXT-	JP NZ NEXT
C9	RET
H	Fim de código
T	Terminar introdução
N	Nomear uma rotina
23784	em 23784
NEXT	Nome da rotina
0	Ponto de entrada relativo

C	Carregar rotina
31000	em 31000
P	Parar o ''Editor de código de máquina''
RUN 9500	Fazer executar a parte em BASIC do código apresentado, que ficou incluída no fim do programa ECM.

Códigos de erro

1 Tentativa para inserir ou apagar código fora do espaço hex
2 A primeira linha não é uma REM (fatal)
3 Declaração a acrescentar à linha 1 não é uma REM (fatal)
4 Tentativa de iniciar código hex fora do espaço hex
5 Tentativa de continuar a escrever código hex para além dc espaço hex
6 Tentativa de rever código hex fora do espaço hex
7 Tentativa de acrescentar um nome fora do espaço hex

Programa 14. *«Editor de código máquina».*

```
    1 REM 111111111111111111111111
1111111111111111111111111111111111
1111111111111111111111111111111111
1111111111111111111111111111111111
111
   20 LET t6=2
   30 LET t24=28
1000 GO TO 2430
1010 LET i=i+1
1030 FOR j=1 TO LEN y$
1040 IF PEEK i=CODE y$(j TO ) TH
EN GO TO 1080
1050 NEXT j
1060 LET b=-1
1070 RETURN
1080 FOR p=1 TO LEN y$-j-2
1090 IF PEEK (i+p)=CODE "," OR P
EEK (i+p)=173 THEN GO TO 1130
1100 IF PEEK (i+p)<>CODE y$(j+p
TO ) THEN GO TO 1050
```

```
1120 NEXT p
1130 LET b=CODE y$(j+p TO )+256*
CODE y$(j+p+1 TO )
1140 LET i=i+p+1-2*(PEEK (i+p)=1
73)
1150 RETURN
1160 GO SUB 1010
1170 IF b>32767 AND b<65536 THEN
 GO TO 1200
1180 PRINT ,,"Erro 11"
1190 STOP
1200 LET y$(j+p TO j+p+1)=CHR$ (
b+k-256*INT ((b+k)/256))+CHR$ (I
NT ((b+k)/256)-128)
1210 PRINT k;"  ";y$(j TO j+p-1)
1220 GO TO 2070
1230 LET b=lo+2*(PEEK (i+1)-49-7
*(PEEK (i+1)>57)-121*(PEEK (i+1)
>127))
1260 POKE k,b-256*INT (b/256)
1270 POKE k+1,INT (b/256)
1280 LET k=k+2
1290 GO TO 2070
1300 LET b=s+PEEK (i+1)-49-7*(PE
EK (i+1)>57)-121*(PEEK (i+1)>127
)
1310 GO TO 1240
1320 GO SUB 1010
1330 GO TO 1240
1370 LET i=5+PEEK 23635+256*PEEK
 23636
1380 LET k=i
1390 PRINT TAB 5;"Carregar codig
o maquina",,,"Int. endereco p/ c
odigo objecto:"
1400 INPUT j: IF j>16384 AND j<P
EEK 23653+256*PEEK 23654 THEN PR
INT ,,"Esta parte da RAM esta' a
 ser u-tilizada. Quer continuar
(S/N)? ": INPUT z$: IF z$="N" OR
 z$="n" THEN PRINT "OK. Indique
outro endereco": GO TO 1400
1480 LET lo=0
1490 LET s=0
1500 GO SUB 6000
1510 IF p=CODE "-" OR p=CODE "."
 THEN GO TO 1540
1515 IF p>CODE "F" THEN GO TO 16
00
1520 LET i=i+1
1530 GO TO 1500
1540 LET i=i+1
```



```
1550 GO SUB 6000
1560 IF p=CODE "-" THEN GO TO 15
20
1570 IF p<>CODE "." THEN GO TO 1
540
1580 LET i=i+3
1590 GO TO 1500
1600 IF p=CODE "H" THEN GO TO 17
50
1602 IF p=CODE "H" THEN GO TO 17
50
1610 LET q=PEEK (i+1)-48-128*INT
 (PEEK (i+1)/128)
1650 IF p=CODE "L" THEN GO TO 16
90
1660 IF p=CODE "S" THEN GO TO 17
20
1690 IF q>lo THEN LET lo=q
1700 LET i=i+2
1710 GO TO 1500
1720 IF q>s THEN LET s=q
1730 LET i=i+2
1740 GO TO 1500
1750 PRINT ,,lo;" variaveis long
as ";s;" variaveis curtas"
1760 LET i=k
1770 LET k=j+2*lo+s
1780 LET ks=k
1790 LET s=2*lo+j
1800 LET lo=j
1810 PRINT "variaveis longas em
";lo,"variaveis curtas em ";s,"p
rograma c.m. comeca em ";k
1830 LET m=i
1840 LET m=i: LET y$=""
1860 GO SUB 6000
1862 IF p=CODE "." THEN GO TO 19
00
1870 IF p=CODE "H" THEN GO TO 19
70
1880 LET i=i+1
1890 GO TO 1860
1900 LET i=i+1
1910 IF PEEK i=CODE "." THEN GO
TO 1940
1920 LET y$=y$+CHR$ PEEK i
1930 GO TO 1900
1940 LET y$=y$+CHR$ PEEK (i+1)+C
HR$ PEEK (i+2)
1950 LET i=i+3
1960 GO TO 1860
1970 LET i=m
```

```
1980 GO SUB 6000: IF p=CODE "H"
THEN PRINT ,,"Fim em ";k-1: RETU
RN
1990 IF p=CODE "," THEN GO TO 11
60
2000 IF p=CODE "L" THEN GO TO 12
30
2010 IF p=CODE "S" THEN GO TO 13
00
2020 IF p=CODE "-" THEN GO TO 13
20
2050 POKE k,16*(PEEK i-7*(PEEK i
>57))-816+PEEK (i+1)-7*(PEEK (i+
1)>57)-121*(PEEK (i+1)>127)-7*(P
EEK (i+1)>185)
2060 LET k=k+1
2070 LET i=i+2
2080 GO TO 1980
2090 PRINT TAB 7;"Descarregar co
digo",,,"Introduza os enderecos
em que   quer comecar e acabar:"
2100 INPUT ks,kf
2110 FOR i=ks TO kf
2130 LET p=INT (PEEK i/16): LET
q=PEEK i-16*INT (PEEK i/16): PRI
NT CHR$ (48+p+7*(p>9));CHR$ (48+
q+7*(q>9));"▒▒";
2140 NEXT i
2150 RETURN
2160 PRINT ,,TAB 9;"Inserir codi
go",,,"Introduza o endereco em q
ue querfazer a insercao e o nume
ro de  elementos a inserir:"
2170 INPUT i,j
2180 IF i<prog OR j<0 OR i+j>pro
g+l THEN BEEP .2,0: PRINT "Erro
1": PAUSE 0: GO TO 2170
2190 FOR k=prog+l-1 TO i+j STEP
-1
2200 POKE k,PEEK (k-j)
2210 NEXT k
2220 GO TO 2700
2340 PRINT ,,TAB 10;"Apagar codi
go",,,"Introduza o endereco em q
ue querapagar e o numero de elem
entos:"
2350 INPUT i,j
2360 IF i<prog OR j<0 OR i>prog+
l THEN BEEP .2,0: PRINT "Erro 1"
: PAUSE 0: GO TO 2350
2370 FOR k=i TO prog+l-j-1
2380 POKE k,PEEK (k+j)
```



```
2390 NEXT k
2400 RETURN
2430 LET prog=5+PEEK 23635+256*P
EEK 23636
2440 LET l=PEEK (prog-3)+256*PEE
K (prog-2)-2
2450 CLS : PRINT TAB 3;"Editor d
e Codigo Maquina"
2460 IF PEEK (prog-1)<>234 THEN
BEEP .2,0: PRINT "Erro 2": STOP
2470 PRINT ,,,,TAB 13;"Menu",,,T
AB t6;"Acrescentar uma REM";TAB
t24;"A";TAB t6;"Carregar uma rot
ina";TAB t24;"C";TAB t6;"Nomear
um segmento";TAB t24;"N";TAB t6;
"Descarregar uma rotina";TAB t24
;"D";TAB t6;"Inserir hex";TAB t2
4;"I";TAB t6;"Apagar hex";TAB t2
4;"X";TAB t6;"Rever hex";TAB t24
;"R";TAB t6;"Escrever hex";TAB t
24;"E";TAB t6;"Parar";TAB t24;"P
",,,"  Espaco hex de ";prog;" a
";prog+l,,,
2480 INPUT z$: IF CODE z$>90 THE
N LET z$=CHR$ (CODE z$-32)
2490 GO SUB 2400+150*(z$="P")+20
0*(z$="A")-1030*(z$="C")+730*(z$
="N")-310*(z$="D")-240*(z$="I")-
60*(z$="X")+500*(z$="R")+280*(z$
="E")
2500 PRINT TAB 0,,"Prima uma tec
la para continuar": PAUSE 0
2510 GO TO 2430
2550 PRINT ,,"O programa parou"
2560 STOP
2610 IF PEEK (prog+l+5)<>234 THE
N BEEP .2,0: PRINT "Erro 3": STO
P
2620 LET l=l+PEEK (prog+l+3)+256
*PEEK (prog+l+4)+6
2630 POKE prog-3,l-256*INT (l/25
6)
2640 POKE prog-2,INT (l/256)
2650 LET l=l-2
2660 RETURN
2680 PRINT ,,TAB 7;"Escrever cod
igo hex",,,"Int. endereco em q.
quer comecar"
2690 INPUT i: IF i<prog OR i>pro
g+l THEN BEEP .2,0: PRINT "Erro
4": PAUSE 0: GO TO 2690
2700 PRINT ,,"Introduza programa
 hex:"
```

```
2710 INPUT z$: IF z$="t" OR z$="
T" THEN RETURN
2720 LET f=0
2730 PRINT i;"  ";
2740 FOR j=1 TO LEN z$
2750 IF i>prog+l THEN BEEP .2,0:
 PRINT "Erro 5": RETURN
2760 LET z$(j)=CHR$ (CODE z$(j)-
32*(CODE z$(j)>96))
2770 POKE i,CODE z$(j)
2780 PRINT z$(j);
2790 LET f=f=1 OR z$(j)="-" OR z
$(J)="."
2800 IF f=0 AND j=2*INT (j/2) TH
EN PRINT " ";
2810 LET i=i+1
2820 NEXT j
2830 PRINT
2840 POKE i-1,PEEK (i-1)+128
2850 GO TO 2710
2900 PRINT ,,TAB 8;"Rever codigo
 hex",,,"Int. endereco em q. que
r comecar"
2910 INPUT i
2920 IF i<prog OR i>prog+l THEN
BEEP .2,0: PRINT "Erro 6": GO TO
 2910
2940 LET j=0
2950 PRINT i;"  ";
2960 GO TO 2970+90*(PEEK i=CODE
".")+80*(PEEK i=CODE "-")
2970 PRINT CHR$ (PEEK i-128*INT
(PEEK i/128));
2975 IF PEEK i=200 THEN PRINT TA
B 0;"Fim de codigo": RETURN
2980 LET i=i+1
2990 LET j=j+1: IF j=2*INT (j/2)
 THEN PRINT ;" ";
3010 IF PEEK (i-1)<128 THEN GO T
O 2960
3020 PRINT
3030 GO TO 2930
3050 PRINT "-";
3060 LET i=i+1
3070 IF PEEK i=173 THEN GO TO 29
70
3080 PRINT CHR$ PEEK i;
3090 IF PEEK (i+1)<>CODE "," THE
N GO TO 3060
3100 PRINT " introducao rel em "
;PEEK (i+2)+256*(PEEK (i+3)-128)
3110 LET i=i+4
```



```
3120 GO TO 2930
3130 PRINT TAB 7;"Nomear um segm
ento",,,"Introduza endereco do s
egmento:"
3140 INPUT i: IF i<prog OR i>pro
g+l THEN BEEP .2,0: PRINT "Erro
7": GO TO 3140
3150 PRINT ,,"Introduza nome:"
3160 INPUT z$
3170 LET z$="."+z$+"."
3180 LET j=LEN z$+2
3190 FOR k=prog+l TO i+j STEP -1
3200 POKE k,PEEK (k-j)
3210 NEXT k
3220 FOR j=1 TO LEN z$
3225 IF CODE z$(j)>96 THEN LET z
$(j)=CHR$ (CODE z$(j)-32)
3230 POKE i+j-1,CODE z$(j)
3240 NEXT j
3250 LET i=i+LEN z$
3260 PRINT ,,"Introduza endereco
 do ponto de  entrada:"
3270 INPUT j
3280 POKE i,j-256*INT (j/256)
3290 POKE i+1,128+INT (j/256)
3300 RETURN
6000 LET p=PEEK i-128*INT (PEEK
i/128)
6010 RETURN
9000 FOR i=0 TO 255
9010 POKE 22528+i,i
9020 NEXT i
9030 PRINT USR 32600
```

CAPÍTULO 5

Programas para tratamento da imagem

GRÁFICOS

Quando começámos a trabalhar neste programa, víamo-lo como um meio de gerar formas que poderiam ser passadas para *cassette* através do uso do comando SCREEN$ e utilizadas posteriormente por outros programas. Contudo, depressa verificámos que, introduzindo algumas pequenas alterações, obteríamos um programa que nos permitiria gerar, armazenar e utilizar os nossos próprios caracteres gráficos e, portanto, é esse programa que apresentamos aqui.

Princípios de funcionamento

Vinte e um dos 256 códigos de carácter do conjunto de caracteres do *Spectrum* (números 144 a 164, inclusive) estão disponíveis para definirmos os nossos próprios caracteres gráficos. A tabela que define a forma dos caracteres reside normalmente no extremo superior da RAM, logo acima do indicador de RAMTOP, como o manual de programação BASIC do *ZX Spectrum* descreve na página 179. Quando ligamos o *Spectrum,* os caracteres gráficos definíveis pelo utilizador são inicializados para uma cópia das letras maiúsculas A a U, e a variável de sistema UDG, com os endereços 23675 e 23676, é fixada de modo a indicar o extremo inferior da área dos caracteres gráficos, com o endereço 32600. Temos acesso aos caracteres gráficos pressionando CAPS SHIFT 9, seguindo-se a letra apropriada e, por fim, novamente CAPS SHIFT 9.

Todos os caracteres do *Spectrum* são construídos a partir de um bloco ou matriz de oito por oito *pixels,* tendo cada *pixel* a possibilidade de ficar ou não iluminado. Esta disposição permite que o padrão de *pixels* que forma um carácter seja armazenado em oito *bytes,* com o estado ou condição de cada um dos oito *bits*

(isto é, 0 ou 1) em cada *byte* definindo se o *pixel* correspondente é ou não iluminado. Deste modo, a condição da primeira linha (de topo) de oito *pixels* num carácter é armazenada no primeiro *byte*,

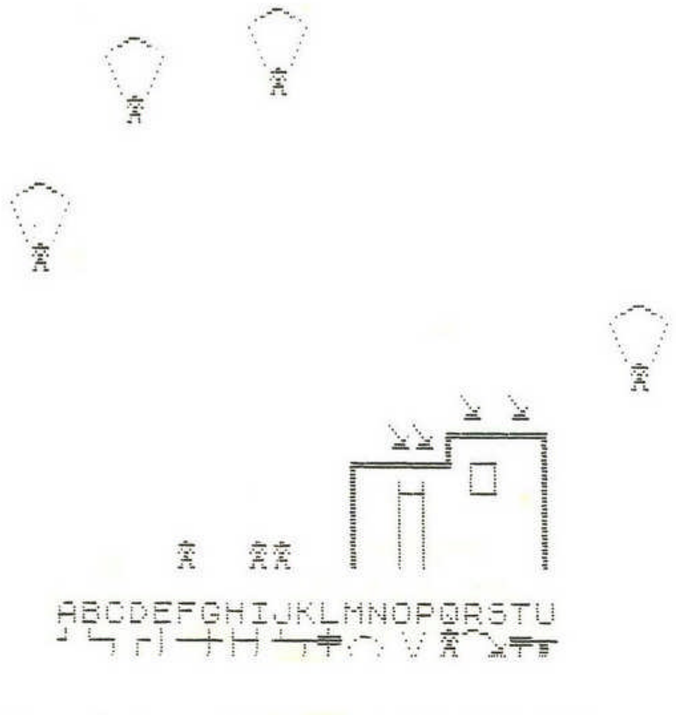


Figura 19. *Exemplo de uma figura construída utilizando o programa «Gráficos».*

a segunda linha de *pixels* é armazenada no segundo *byte* e assim por diante. A forma de 21 caracteres diferentes tem de ser, obviamente, armazenada numa tabela de $21 \times 8 = 168$ *bytes*. Por isso, não é motivo de surpresa observar que o indicador (UDG) é fixado em 32600, isto é, 168 *bytes* abaixo do extremo superior da RAM (no *Spectrum* de 16 k).

Se introduzirmos, por meio de POKE, um novo valor na variável de sistema UDG, a área de memória a que temos acesso ao construir caracteres gráficos definíveis pelo utilizador mudará e, por isso, os caracteres mudarão também. Vê-se este efeito na rotina seguinte, a qual imprime uma linha de caracteres gráficos, reajusta o valor de UDG, imprime seguidamente outra linha, etc.

Programa 15. *Este programa imprime estranhos caracteres gráficos.*

```
10 FOR i=1 TO 21
20 PRINT AT i,0;"ABCDEFGHIJKLM
NOPQRSTUABCDEFGHIJK"
30 LET j=RND*32768
40 POKE 23675,j-256*INT (j/256
)
50 POKE 23676,INT (j/256)
60 NEXT i
70 GO TO 10
```

A linha de caracteres na declaração PRINT da linha 20 deve ser precedida e seguida por CAPS SHIFT 9 quando for introduzida pelo teclado, pois esses caracteres devem ser introduzidos em modo gráfico.

Acerca do programa

Quando é executado, o programa desloca RAMTOP para baixo, para o endereço 31087, de forma a deixar espaço para dez conjuntos de caracteres definidos pelo utilizador, contendo cada conjunto 21 caracteres. Seguidamente, introduz uma rotina de inicialização (linhas 500 a 900), a qual armazena 32 caracteres destinados à elaboração de formas no primeiro conjunto de caracteres e em parte do segundo. Os dados para estes caracteres são lidos nas declarações DATA e vemos nas declarações PRINT das linhas 760, 1060 e 2610 alguns dos caracteres resultantes. Pode voltar a ter-se acesso a esta rotina inicializante a partir do *menu* principal, introduzindo I.

O *menu* principal, tal como é apresentado na figura 20, é impresso no *écran* quando se completa a inicialização. A metade superior do *écran* é uma grelha de oito por oito posições de impressão, com a qual se constroem interactivamente novos caracteres, se apresentam caracteres quer do conjunto de caracteres normal quer dos conjuntos de caracteres definidos pelo utilizador e se transferem caracteres de uma opção para outra. Esta última particularidade permite copiar caracteres já existentes e modificá-los de forma a criar novos caracteres.

Quando se selecciona a opção «Acrescentar caracteres ao conjunto», o programa pergunta-nos se desejamos «limpar» o *écran* e «convida-nos» seguidamente a introduzir pares de coordenadas de forma a criar ou alterar o carácter no *écran*. Após ter sido introduzido cada par de coordenadas, o *software* verifica os atributos da célula de impressão correspondente e, se esta estiver vazia, imprime o caracter gráfico número oito (da tecla 8) e reacerta os valores de INK e PAPER («tinta» e «fundo»). É este dispositivo que permite imprimir sobre caracteres existentes sem os apagar, como aconteceria se fosse utilizada a função OVER.

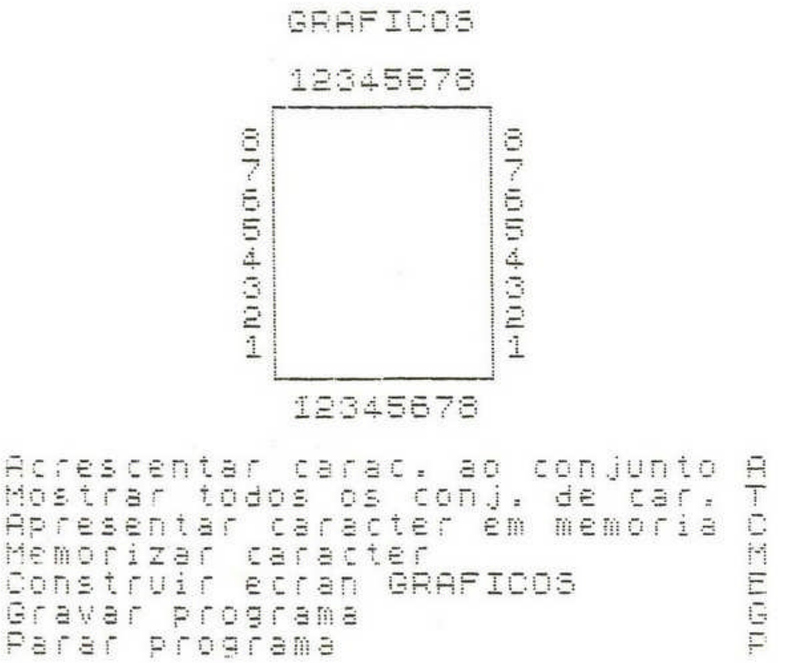


Figura 20. *O* menu *principal de «Gráficos».*

A opção «Mostrar todos os conjuntos de caracteres» limpa a totalidade do *écran* e imprime todos os caracteres gráficos definidos pelo utilizador. A opção «Apresentar carácter em memória» copia um carácter de qualquer dos dez conjuntos de caracteres definidos pelo utilizador ou do conjunto de caracteres normais (conjunto número zero) da memória para a grelha da metade superior do *écran,* enquanto a opção «Memorizar carácter» efectua a função inversa.

Tanto o programa como as variáveis e os conjuntos de caracteres podem ser gravados em *cassette* utilizando a opção «Gravar programa». Contudo, antes de efectuar a gravação, deve-se introduzir

CLEAR 31087

pelo teclado para reajustar a RAMTOP.

Construir um «écran»

Depois de criarmos caracteres adequados, tê-los armazenado e, possívelmente, gravado em *cassette* utilizando o *menu* principal, podemos seleccionar o ambiente «Construir *écran* gráfico» para elaborar um desenho à escolha. Neste ambiente, temos à nossa disposição 20 linhas por 32 colunas, servindo as linhas 20 e 21 para apresentar o conjunto de caracteres a utilizar no momento. A imagem pode ser construída com todos os caracteres dos dez conjuntos de definição pelo utilizador e com os caracteres do conjunto normal de caracteres. Podemos gravar o desenho de um *écran* em *cassette* para o utilizarmos em outro programa. Do mesmo modo, podemos copiar para o *écran* um desenho de um outro programa.

Quando seleccionamos este ambiente, as várias opções disponíveis são apresentadas no *écran,* como mostra a figura 21.

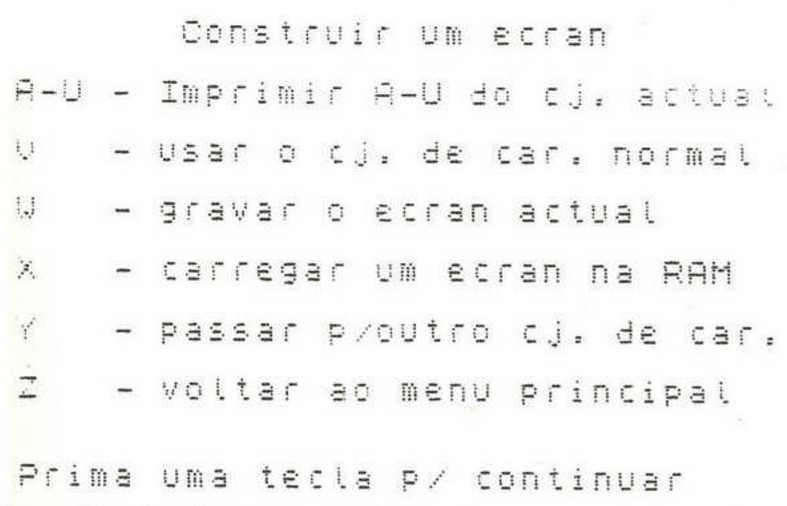


Figura 21. *As várias opções disponíveis para construir um* écran. (Nota: Cj significa «conjunto»)

Apenas são permitidas introduções de caracteres entre A e U, excepto quando se utiliza o conjunto normal de caracteres. A introdução de um carácter do grupo «A» a «U», seguido por um par de coordenadas, faz com que o carácter correspondente do conjunto de caracteres definidos pelo utilizador a utilizar no momento seja impresso no *écran,* na posição definida pelas coordenadas. Podemos repetir a introdução de caracteres tantas vezes quantas as desejadas para se construir a forma, imagem, mapa, etc., pretendida. Podemos seleccionar outro conjunto de caracteres, sem alterar o conteúdo do *écran,* introduzindo «Y». O novo conjunto seleccionado é apresentado então na parte inferior do *écran*. O conjunto de caracteres normal é seleccionado de forma semelhante introduzindo «V». Com as letras «W» e «X» grava-se e carrega-se, em memória respectivamente, um *écran* enquanto que «Z» provoca um regresso ao *menu* principal.

«Spectrum» de 48 k

O *Spectrum* de 48 k pode executar o programa na forma que aqui apresentamos. O espaço restante acima de RAMTOP pode ser utilizado para armazenar cópias das imagens do *écran* desenhadas com a opção descrita anteriormente. As duas rotinas seguintes copiam a imagem no *écran* para 40000 e deste endereço novamente para o *écran*. Com o programa «Carregador de código máquina» apresentamos uma rotina em código máquina destinada a executar a mesma função.

```
9000 FOR i=0 TO 6143
9010 POKE 40000+i, PEEK (16384+i)
9020 NEXT i
9050 FOR i=0 TO 6143
9060 POKE 16384+i, PEEK (40000+i)
9070 NEXT i
```

Programa 15-A. *«Gráficos»*

```
  40 REM "GRAFICOS"
  50 CLEAR 31087
  60 DIM u(10)
  70 FOR i=0 TO 9
  80 LET u(i+1)=31088+168*i
  90 NEXT i
 100 BORDER 1: PAPER 1: INK 7
 110 GO TO 500
 150 GO SUB 8800
 155 BORDER 1: PAPER 1: INK 7
 160 PRINT AT 15,0;"Acrescentar
carac. ao conjunto";TAB 31;"AMos
trar todos os conj. de car. TApr
esentar caracter em memoria CMem
orizar caracter";TAB 31;"MConstr
uir ecran GRAFICOS          EGravar
 Programa";TAB 31;"GParar progra
ma";TAB 31;"P"
 170 INPUT z$: LET z$=CHR$ (CODE
 z$-32*(z$>"Z"))
 180 GO TO 200+300*(z$="I")+800*
(z$="T")+1300*(z$="C")+1800*(z$=
"M")+2300*(z$="E")+2800*(z$="G")
+3300*(z$="P")+3800*(z$="A")
 200 INPUT "Prima ENTER para con
tinuar";z$
 210 GO TO 150
 500 DATA 3,16,1,240,4,0,3,16,1,
31,7,0,1,240,4,16,3,0,1,31,12,16
,3,0,1,255,4,0,3,16,1,255,7,16,1
,31,7,16,1,240,7,16,1,255,7,0,1,
255,4,16
 510 DATA 2,16,3,255,3,16,3,56,1
,255,7,56,1,63,4,56,2,16,3,31,5,
16,3,255,3,0,3,56,1,255,4,0,3,56
,1,248,4,56,2,16,3,240,3,16,2,0,
3,255,3,16,3,0,1,255,4,56
 520 DATA 3,24,2,248,3,0,3,24,2,
31,6,0,2,248,3,24,3,0,2,31,11,24
,3,0,2,255,3,0,3,24,2,255,6,24,2
,31,6,24,2,248,6,24,2,255,3,24,3
,0,2,255,3,24
 590 GO SUB 8500
 600 RESTORE
 610 PRINT TAB 2; FLASH 1;"Inici
alizacao a ser efectuada"
 660 PRINT AT 20,5;"ABCDEFGHIJKL
MNOPQRSTU";TAB 5;
 690 POKE 23675,112: POKE 23676,
121
 700 LET n=0
 710 FOR i=0 TO 255
 720 IF n=0 THEN READ n,k
 730 LET n=n-1
 740 POKE 31088+i,k
```

```
 750 NEXT i
 760 PRINT "┘ ┐ ┌│─├┤┤ ┴┬╪╫╞═╧╣╫╤
╥"
 800 FOR i=31344 TO 32767
 810 POKE i,0
 820 NEXT i
 900 GO TO 150
1000 CLS : PRINT TAB 12;"Grafico
s"
1010 PRINT ,,"Mostrar todos os c
onj. de carac."
1020 PRINT AT 4,5;"ABCDEFGHIJKLM
NOPQRSTU",,TAB 5;
1030 FOR i=31088 TO 32600 STEP 1
68
1040 POKE 23675,i-256*INT (i/256
)
1050 POKE 23676,INT (i/256)
1060 PRINT "┘ ┐ ┌│─├┤┤ ┴┬╪╫╞═╧╣╫╤
╥";TAB 5;
1070 NEXT i
1080 PRINT
1090 INPUT "Prima ENTER para con
tinuar ";z$
1100 GO SUB 8500
1110 GO TO 150
1200 GO TO 200
1500 GO SUB 8800
1510 PRINT " Apresentar caracter
 em memoria"
1520 INPUT "Limpar ecran (S ou N
) ?";z$
1530 LET z$=CHR$ (CODE z$-32*(z$
>"£"))
1540 IF z$<>"S" AND z$<>"N" THEN
 BEEP .2,24: GO TO 1520
1550 IF z$="S" THEN GO SUB 8500
1560 LET im=0
1570 GO SUB 8000
1580 IF i<>0 THEN GO TO 1700
1600 LET r=4: LET c=12
1610 FOR j=0 TO 7
1620 LET u=PEEK (15616+8*(CODE z
$-32)+j)
1630 GO SUB 1900
1640 NEXT j
1650 GO TO 200
1710 LET r=4: LET c=12
1720 FOR j=0 TO 7
1730 LET u=PEEK (u(i)+j+8*(CODE
z$-65))
1740 GO SUB 1900
1750 NEXT j
1790 GO TO 200
1900 FOR k=0 TO 7
1910 LET t=2↑(7-k)
1920 IF INT (u/t)=1 AND ATTR (r+
j,c+k)=15 THEN PRINT INK 1; PAPE
R 7; INVERSE 1;AT r+j,c+k;"█"
1930 LET u=u-t*INT (u/t)
1940 NEXT k
1950 RETURN
2000 GO SUB 8800
2010 PRINT TAB 7;"Memorizar cara
cter"
2020 LET im=0: LET r=4: LET c=12
2030 GO SUB 8000
2040 PRINT FLASH 1;"Espere por f
avor"
2060 FOR j=0 TO 7
2070 LET u=0
2080 FOR k=0 TO 7
2090 IF ATTR (r+j,c+k)<>15 THEN
LET u=u+2↑(7-k)
2100 NEXT k
2110 POKE u(i)+8*(CODE z$-65)+j,
u
2120 NEXT j
2200 GO TO 200
2500 CLS : PRINT TAB 7;"Construi
r um ecran"
2520 PRINT ,,"A-U - Imprimir A-U
 do cj. actual"
2530 PRINT ,,"V   - usar o cj. d
e car. normal"
2540 PRINT ,,"W   - gravar o ecr
an actual"
2550 PRINT ,,"X   - carregar um
ecran na RAM"
2560 PRINT ,,"Y   - passar p/out
ro cj. de car."
2570 PRINT ,,"Z   - voltar ao me
nu principal"
2580 PRINT ,,,,"Prima uma tecla
p/ continuar"
2590 PAUSE 0
2600 CLS : LET i=1: POKE 23675,1
12: POKE 23676,121
2610 PRINT AT 20,5;"ABCDEFGHIJKL
MNOPQRSTU";TAB 5;"┘ ┐ ┌│─┼┤ ┴┬╪╫
╞═╧╣╤╥"
2620 INPUT z$: LET z$=CHR$ (CODE
 z$-32*(z$>"£")): IF z$<"A" OR z
$>"Z" THEN BEEP .2,24: GO TO 261
0
```

```
2630 GO TO 2700+3300*(z$<="U")+3
400*(z$="V")+3500*(z$="W")+3600*
(z$="X")+3700*(z$="Y")
2700 GO SUB 8500
2710 GO TO 150
3000 GO SUB 8800
3010 PRINT TAB 8;"Gravar GRAFICO
S"
3020 PRINT ,,"Int. nome p/ fiche
iro em cass.:"
3030 INPUT z$: IF LEN z$>10 THEN
 BEEP .2,24: GO TO 3030
3040 SAVE z$ LINE 3100
3050 PRINT ,,"Caracteres de GRAF
ICOS a serem  gravados"
3060 SAVE z$CODE 31088,1680
3070 PRINT ,,"Gravacao terminada
"
3080 GO TO 200
3100 LOAD z$CODE 31088,1680
3110 GO TO 200
3500 GO SUB 8800
3510 PRINT "O programa parou"
3520 STOP
4000 GO SUB 8800
4010 PRINT "Acrescentar caracter
 ao conjunto"
4020 INPUT "Limpar ecran (S ou N
) ?";z$
4030 LET z$=CHR$ (CODE z$-32*(z$
>"£"))
4040 IF z$<>"S" AND z$<>"N" THEN
 BEEP .2,24: GO TO 4020
4050 IF z$="S" THEN GO SUB 8500
4060 INPUT "Int. coord. (1 a 8),
 0 p/ term. ";a,b
4070 IF a<0 OR a>8 OR b<0 OR b>8
 THEN BEEP .2,24: GO TO 4060
4080 IF a=0 OR b=0 THEN GO TO 20
0
4090 IF ATTR (12-a,11+b)=15 THEN
 PRINT INK 1; PAPER 7; INVERSE 1
;AT 12-a,11+b;"■"
4100 GO TO 4060
6000 INPUT "Intr. X,Y ";x,y
6010 IF x<0 OR y<0 OR x>19 OR y>
31 THEN BEEP .2,24: GO TO 6000
6020 PRINT AT x,y;CHR$ (CODE z$+
79)
6030 GO TO 2610
6100 INPUT "Int. car. ,X,Y ";z$;
" ";x;" ";y
```



```
6110 IF z$<" " OR z$>"©" OR x<0
OR y<0 OR x>19 OR y>31 THEN BEEP
 .2,24: GO TO 6100
6120 IF z$=" " THEN GO TO 2610
6130 PRINT AT x,y;z$
6140 GO TO 6100
6200 INPUT "Int. nome p/ o ecran
";z$
6210 IF LEN z$>10 THEN BEEP .2,2
4: GO TO 6200
6230 SAVE z$SCREEN$
6240 GO TO 2610
6300 INPUT "Intr. nome do ecran
";z$
6310 IF LEN z$>10 THEN BEEP .2,2
4: GO TO 6300
6320 LOAD z$SCREEN$
6340 GO TO 2610
6400 INPUT "Int. num. do conj. d
e car. ";i
6410 IF i<1 OR i>10 THEN BEEP .2
,24: GO TO 6400
6420 POKE 23675,u(i)-256*INT (u(
i)/256)
6430 POKE 23676,INT (u(i)/256)
6440 GO TO 2610
8000 INPUT "Int. num. do conj. d
e car. ";i
8005 LET a$="A": LET b$="U"
8010 IF i<im OR i>10 THEN BEEP .
2,24: GO TO 8000
8020 IF i<>0 THEN GO TO 8050
8030 INPUT "Intr. caracter (  a
©) ";z$
8040 GO TO 8050
8050 INPUT "Intr. caracter (A a
U) ";z$
8060 IF i=0 THEN LET a$=" "
8070 IF i=0 THEN LET b$="©"
8080 IF z$<a$ OR z$>b$ THEN BEEP
 .2,24: GO TO 8020
8090 IF i=0 THEN RETURN
8100 POKE 23675,u(i)-256*INT (u(
i)/256)
8110 POKE 23676,INT (u(i)/256)
8120 RETURN
8500 CLS : PRINT TAB 12;"GRAFICO
S"
8510 PRINT AT 2,12;"12345678"
8520 FOR i=1 TO 8
8530 PRINT AT 3+i,10;9-i;TAB 21;
9-i
```

```
8540 NEXT i
8550 PRINT AT 13,12;"12345678"
8560 PLOT 91,75
8570 DRAW 73,0
8580 DRAW 0,73
8590 DRAW -73,0
8600 DRAW 0,-73
8700 RETURN
8800 FOR i=14 TO 21
8810 PRINT AT i,0;"
                  "
8820 NEXT i
8830 PRINT AT 14,0;
8840 RETURN
```

DESENHO

Escrevemos este programa logo após ter completado o programa «Música» e, devido a isso, há uma série de semelhanças entre eles. São ambos controláveis por *menus* e ambos armazenam os dados que determinam a música/desenho num quadro numérico. Em ambos os casos, podemos escrever e reescrever os dados, experimentá-los e alterá-los até ficarmos satisfeitos. Podemos então gravar separadamente em *cassette* a melodia ou o desenho resultante, para ser utilizado em ocasião posterior.

Todas as funções de desenho deste programa utilizam o comando DRAW, descrito no capítulo 17 do manual de programação em BASIC do *ZX Spectrum*. O formato básico do comando é:

DRAW x,y

onde x e y são os deslocamentos relativos, horizontal e vertical, a efectuar-se a partir da posição de traçagem *(plotting position)* corrente. Uma variante do comando é:

DRAW x,y,a

onde x e y são os mesmos que anteriormente e a é o valor do ângulo, em radianos, que a linha deve rodar ao efectuar o deslocamento. Ao utilizar este comando no programa, convertemos todos os ângulos de radianos para graus, porque pensamos

que muitas pessoas estão mais familiarizadas com esta forma de medida de ângulos.

Quando o fazemos executar, o programa pede-nos as cores que desejamos para a margem (BORDER) e para o fundo (PAPER), e ainda o número de passos que pensamos introduzir. Seguidamente, define um quadro bidimensional que conterá a seguinte informação:

- b(i,1) Deslocamento relativo na direcção x
- b(i,2) Deslocamento relativo na direcção y
- b(i,3) Ângulo, em graus, que a linha deve rodar
- b(i,4) −1 para «caneta» levantada, 0−7 para determinar a cor da linha a desenhar.
- b(i,5) Coordenada x absoluta corrente
- b(i,6) Coordenada y absoluta corrente

Controlamos o programa por meio do *menu* que vemos na figura 22, o qual é gerado na linha 200. A introdução de uma das letras da coluna da direita faz com que o programa execute a rotina correspondente, para a qual «salta» por meio da linha 230.

```
Desenho do Spectrum

        Menu

Escrever          E
Rever             R
Imprimir          I
Desenhar          D
Fundo             F
Carregar          C
Gravar            G
Verificar         V
Parar             P

O programa parou
```

Figura 22. *O* menu *do «Desenho do* Spectrum».

Escrever Introduzir E

Esta opção permite-nos escrever novos passos ou reescrever passos que queiramos corrigir. O programa pede o passo em que desejamos começar — normalmente 1.

O programa começa sempre o desenho no canto inferior esquerdo do *écran*, nas coordenadas x=0, y=0. Cada passo subsequente é iniciado no ponto em que terminou o passo anterior. A distância máxima a que é possível efectuar um deslocamento é, no total, 225 unidades na direcção x (horizontal) e 175 unidades na direcção y. O programa não aceita valores que desloquem a «caneta» de desenho para além desses limites, mas devemos ter o cuidado de verificar que as curvas que definimos também se mantêm dentro dos limites indicados, pois de contrário o programa parará com uma mensagem de erro.

O programa pede, à vez, a introdução dos valores da deslocação x e da deslocação y, e ainda o valor do ângulo, em graus, do arco de circunferência que a posição de traçagem deve efectuar no sentido anti-horário. Um ângulo negativo fará desenhar no sentido horário (dos ponteiros do relógio) o arco de circunferência. Seguidamente, o programa solicita a cor em que a linha deve ser desenhada e aceita valores no intervalo 0 a 7, os quais definem a cor indicada por cima da correspondente tecla do *Spectrum*. Se introduzirmos aqui L, a «caneta» de desenho «levantar-se-á» durante o período do passo, permitindo-nos assim deslocar a posição de traçagem de um ponto do desenho para outro sem que seja desenhada uma linha.

Rever e imprimir Introduzir R ou I

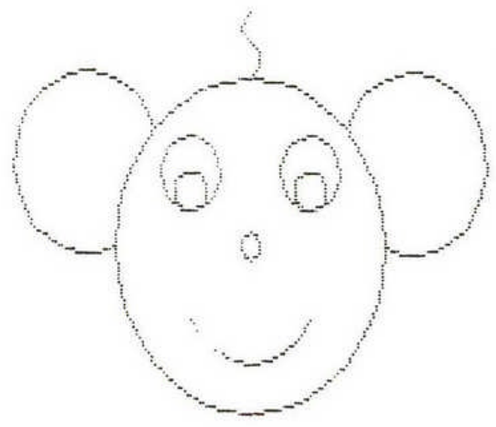
Estas duas opções listam as instruções em memória no *écran* ou através da impressora. A tabela 8 apresenta a listagem das instruções necessárias para desenhar o «rato» da figura 23.

Tabela 8. *Instruções típicas usadas no «Desenho».*

```
Imprimir instrucoes

RATO SORRIDENTE

Passo      X        Y      Graus   L/D
  1      127       40        0      L
  2        0       90      180      D
  3        0      -90      180      D
  4      -45       45        0      L
  5     13.2     31.8     -270      4
  6     76.8    -31.8        0      L
  7    -13.2     31.8     -270      4
  8    -11.6    -21.8        0      L
  9        0       20      180      D
 10        0      -20      180      D
 11      -40        0        0      L
 12        0       20      180      D
 13        0      -20      180      D
 14        0       10      180      D
 15        0      -10      180      D
 16       40        0        0      L
 17        0       10      180      D
 18        0      -10      180      D
 19        0      -30        0      L
 20      -40        0     -120      4
 21       20       16        0      L
 22        0        8      150      D
 23        0       -8      150      D
 24        0      -49        0      L
 25        0      -10      135      D
 26        0        8     -130      4
```



RATO SORRIDENTE

Figura 23. *Um desenho típico.*

Fundo Introduzir F

Esta opção permite-nos alterar as cores do fundo e da margem do desenho. De notar a utilização da cor 9 (contraste) para a «tinta» (INK), de forma a que todos os caracteres apresentados no *écran* sejam legíveis, qualquer que seja a cor do fundo.

Desenhar Introduzir D

Nesta opção o programa executa as instruções que se encontrem em memória para traçar o desenho no écran.

Programa 16. *«Desenho»*.

```
  5 REM "DESENHO"
 10 CLS : PRINT TAB 6;"Desenho
do Spectrum",,,"Introduza:"'"
Numero de passos"'"  Cor da marg
em (0 a 7)","  Cor do fundo (0 a
 7)"
 20 INPUT np,m,f: IF np<0 OR m<
0 OR m>7 OR f<0 OR f>7 THEN BEEP
 .2,0: GO TO 10
 30 BORDER m: PAPER f: INK 9
 40 LET fim=0
 50 DIM b(np+1,6): DIM b$(32)
 60 LET b(1,1)=np
 70 LET b(1,2)=m
 80 LET b(1,3)=f
 90 LET b(1,5)=0
100 LET b(1,6)=0
200 CLS : PRINT TAB 6;"Desenho
do Spectrum",,,TAB 14;"Menu",,,T
AB 6;"Escrever";TAB 24;"E",TAB 6
;"Rever";TAB 24;"R",TAB 6;"Impri
mir";TAB 24;"I",TAB 6;"Desenhar"
;TAB 24;"D",TAB 6;"Fundo";TAB 24
;"F",TAB 6;"Carregar";TAB 24;"C"
,TAB 6;"Gravar";TAB 24;"G",TAB 6
;"Verificar";TAB 24;"V",TAB 6;"P
arar";TAB 24;"P"
210 INPUT z$
220 IF z$>="a" THEN LET z$=CHR$
(CODE z$-32)
230 GO TO 300+200*(z$="E")+700*
(z$="F")+1200*(z$="R")+1700*(z$=
"C")+2200*(z$="G")+2700*(z$="D")
+3200*(z$="V")+3700*(z$="P")+420
0*(z$="I")
300 PAUSE 200
310 GO TO 200
500 PRINT ,,"Escrever ou corrig
ir instrucoes"'"Introduza o num
ero do passo em  que deseja come
car a escrever:"
510 INPUT j: IF j<1 OR j>np THE
N GO TO 510
520 PRINT ,,"Para cada passo in
troduza, um a um, os valores da
deslocacao em x e em y, o do ang
ulo, e o da  cor ou L (caneta l
evantada)."'"Introduza:"'"  T
para terminar"'"  V para voltar
atras"
540 PRINT ,," Passo   X     Y
  Graus L/C",,
550 FOR i=j+1 TO np+1
560 PRINT AT 21,3;i-1;TAB 8;"▒▒
▒▒": INPUT x$: IF x$="t" OR x$="
T" THEN GO TO 900
570 IF x$="v" OR x$="V" THEN BE
EP .2,0: BEEP .2,0: LET i=i-1: G
O TO 560
575 PRINT AT 21,8;x$;"     ";: L
ET b(i,5)=b(i-1,5)+VAL x$: IF b(
i,5)>255 OR b(i,5)<0 THEN BEEP .
2,0: GO TO 560
580 PRINT AT 21,15;"▒▒▒▒▒";: INP
UT y$: IF y$="v" OR y$="V" THEN
BEEP .2,0: GO TO 560
585 PRINT AT 21,15;y$;"     ";:
LET b(i,6)=b(i-1,6)+VAL y$: IF b
(i,6)>175 OR b(i,6)<0 THEN BEEP
.2,0: GO TO 580
590 PRINT AT 21,22;"▒▒▒▒▒": INPU
T z$: IF z$="v" OR z$="V" THEN B
EEP .2,0: GO TO 580
595 PRINT AT 21,22;z$;"    ";
600 PRINT AT 21,28;"▒";: INPUT
a$: IF a$="v" OR a$="V" THEN BEE
P .2,0: GO TO 590
605 PRINT AT 21,28;a$
610 IF a$<>"l" AND a$<>"L" AND
(a$<"0" OR a$>"7") THEN BEEP .2,
0: GO TO 600
620 LET b(i,1)=VAL x$: LET b(i,
2)=VAL y$: LET b(i,3)=VAL z$: LE
T b(i,4)=-1*(a$="l" OR a$="L")+(
CODE a$-48)*(a$<>"l" AND a$<>"L"
)
650 PRINT
660 NEXT i
670 PRINT ,,"Nao ha' mais espac
o"
680 LET fim=np+1
690 GO TO 300
900 IF i>fim THEN LET fim=i-1
910 GO TO 200
1000 PRINT ,,TAB 5;"Mudar a cor
do fundo",,,"Os valores actuais
```

```
sao:"//"  Margem",m,"  Fundo",f,
,,"Introduza novos valores:"
1010 INPUT m,f
1020 BORDER m
1030 PAPER f
1040 GO TO 70
1500 PRINT ,,TAB 7;"Rever instru
coes",,," Passo    X       Y     Gr
aus   L/C"
1510 FOR i=2 TO fim
1520 PRINT TAB 3;i-1;TAB 8;b(i,1
);TAB 15;b(i,2);TAB 22;b(i,3);TA
B 29;CHR$ (68+8*(b(i,4)=-1)-(20-
b(i,4))*(b(i,4)>=0))
1530 NEXT i
1540 PRINT ,,"Prima uma tecla p/
voltar ao menu"
1550 PAUSE 32768
1560 GO TO 200
2000 PRINT ,,TAB 6;"Carregar um
desenho";AT 21,0;"Introduza o no
me do desenho:"
2010 INPUT z$: IF LEN z$>10 THEN
 PRINT AT 21,0;"O nome e' longo d
emais. Corrija-o": GO TO 2010
2020 PRINT AT 21,0;"  ";AT 21,2;
 FLASH 1;"Carregamento a ser efe
ctuado";TAB 30
2030 LOAD z$ DATA b()
2040 LET np=b(1,1)
2050 LET m=b(1,2)
2060 LET f=b(1,3)
2070 LET fim=b(1,4)
2080 BORDER m
2090 PAPER f
2100 PRINT AT 19,0;z$;" em memor
ia";TAB 30
2110 GO TO 300
2500 PRINT ,,TAB 7;"Gravar um de
senho";AT 21,0;"Introduza o nome
."
2510 INPUT z$: IF LEN z$>10 THEN
 PRINT AT 21,0;"O nome e' longo d
emais. Corrija-o": GO TO 2510
2520 LET b(1,4)=fim
2530 PRINT AT 21,0;"    ";AT 21,
4; FLASH 1;"Gravacao a ser efect
uada"
2540 SAVE z$ DATA b()
2550 PRINT AT 21,0;z$;" encontra
-se gravado";TAB 30
2560 GO TO 300
```



```
3000 PRINT AT 21,0;"Deseja copia
 impressa (S ou N) ?": INPUT k$:
 IF k$="s" OR k$="S" THEN GO SUB
 5000: IF k$<>"s" AND k$<>"S" AN
D k$<>"n" AND k$<>"N" THEN BEEP
.2,0: GO TO 3000
3005 CLS : PRINT AT 21,5; FLASH
1;"Desenho a ser executado"
3010 PAUSE 100
3020 FOR i=2 TO fim
3025 IF b(i,4)=-1 THEN GO TO 303
5
3030 DRAW INK b(i,4);b(i,1),b(i,
2),b(i,3)*PI/180
3035 PLOT b(i,5),b(i,6)
3040 NEXT i
3045 IF k$="s" OR k$="S" THEN PR
INT AT 21,0;b$;AT 21,15-LEN z$/2
;z$: COPY
3050 PRINT AT 21,0; FLASH 1;"Des
enho acabado. Prima uma tecla"
3060 PAUSE 0
3070 GO TO 200
3500 PRINT ,,TAB 6;"Verificar um
 desenho";AT 21,0;"Introduza o n
ome do desenho:"
3510 INPUT z$: IF LEN z$>10 THEN
 BEEP .2,0: PRINT AT 21,0;"O nom
e e' longo demais. Corrija-o": GO
 TO 3510
3520 PRINT AT 21,0;"  ";AT 21,2;
 FLASH 1;"Verificacao a ser efec
tuada"
3530 VERIFY z$ DATA b()
3540 PRINT AT 19,0;z$;" verifica
do: OK";TAB 30
3550 GO TO 300
4000 PRINT ,,"O programa parou"
4010 STOP
4500 GO SUB 5000: PRINT AT 21,0;
"    ";AT 21,3; FLASH 1;"Impressa
o a ser efectuada";TAB 30: LPRIN
T ,,TAB 6;"Imprimir instrucoes"'
'TAB 15-LEN z$/2;z$''" Passo    X
      Y      Graus   L/C"
4510 FOR i=2 TO fim
4520 LPRINT TAB 3;i-1;TAB 8;b(i,
1);TAB 15;b(i,2);TAB 22;b(i,3);T
AB 29;CHR$ (68+8*(b(i,4)=-1)-(20
-b(i,4))*(b(i,4)>=0))
4530 NEXT i
4540 GO TO 200
```

```
5000 CLS : PRINT AT 21,0;"Introd
uza o nome do desenho :"
5010 INPUT z$: RETURN
```

SPIROMANIA

«Spiromania» é um programa destinado a dar vazão ao talento, aos dotes artísticos e à habilidade do possuidor do *Spectrum*. Não existe limite para a variedade de curvas e de espirais que podemos desenhar com este programa, com excepção das impostas pela nossa imaginação. O programa apresenta opções que nos permitem seleccionar e fazer desenhar uma de oito curvas apresentadas como exemplo, experimentar o desenho de uma curva da nossa autoria e, quando ficarmos satisfeitos, armazenar essa curva em memória e gravá-la para uma ocasião futura.

Estrutura do programa

Quando o fazemos executar, o programa define uma cadeia de caracteres f$, na qual se guardarão de modo definitivo as equações a representar graficamente. Depois, lê oito pares de equações de exemplo nas declarações DATA das linhas 200 a 270 e transfere-as para f$ por intermédio das cadeias de armazenamento temporário x$ e y$.

O programa tem igualmente de armazenar o comprimento, em caracteres, de cada equação, de modo a poder extraí-las correctamente de f$. Poderia efectuá-lo pelo método de definir um quadro numérico e memorizar o comprimento de cada equação nos elementos do quadro. Este método apresenta, contudo, uma desvantagem, pois, no caso de ser utilizado, haveria duas limitações ao número de equações que o programa poderia armazenar, sendo a primeira limitação o comprimento de f$ e a segunda o número de elementos do quadro. Para evitar este problema, o programa reserva em f$ o *byte* que precede cada equação para armazenar o comprimento da equação medido em caracteres. Deste modo, o comprimento de f$ torna-se a única limitação à capacidade do programa e esse comprimento pode ser

facilmente aumentado incrementando o valor de t1, atribuído na linha 110. A única desvantagem desta técnica é que o comprimento de uma equação é armazenada num único *byte* e, portanto, fica limitado a um máximo de 255 caracteres.

Tendo armazenado as equações de exemplo em f$, o programa imprime o *menu* apresentado na figura 24. Se preferirmos visionar uma das curvas de exemplo, devemos seleccionar a opção «Desenhar uma curva». O programa imprime, nesta opção, a lista das equações que se encontrem armazenadas nesse momento em f$ e podemos então escolher uma delas. O programa transfere seguidamente as equações para x$ e y$, entra numa sub-rotina na linha 1600 e desenha a curva seleccionada, continuando a fazê-lo até que pressionemos qualquer tecla para voltar ao *menu*.

```
Spiromania

Menu

Experimentar uma funcao          E
Desenhar uma curva               D
Memorizar uma funcao             M
Gravar e verificar o prog.       G
Parar o programa                 P

O programa parou
```

Figura 24. *O* menu *de «Spiromania».*

Podemos experimentar um novo par de equações não incluído nos de exemplo seleccionando a opção «Experimentar uma função». O computador não verifica a sintaxe das equações e, portanto, devemos ter o cuidado de nos certificarmos de que nelas não existem quaisquer caracteres ilegais (variáveis com nomes diferentes de n, por exemplo), que todos os parênteses têm par, etc. Se cometermos algum erro, o programa pára e imprime a mensagem de erro «C Nonsense in Basic, 1740:1», onde «Nonsense in Basic» se poderá traduzir por «Não tem sentido em Basic», sendo «1740:1» o código para «linha 1740, declaração 1». Esta situação pode ser recuperada introduzindo GOTO 1000 para obter o *menu* e seleccionando, seguidamente, de novo a opção «Experimentar uma função».

As equações que introduzimos são armazenadas temporariamente em x$ e y$, de forma que, se introduzirmos um segundo par de equações, se perde o par anterior. Para reter um par de equações, devemos seleccionar a opção «Memorizar uma função». Isto faz com que o programa transfira o conteúdo corrente de x$ e de y$ para f$, de modo a que a nova curva fique permanentemente acessível por meio da opção «Desenhar uma função». Assim, podemos elaborar uma colecção de curvas experimentando equações, alterando-as até ficarmos satisfeitos com os resultados e, então, armazená-las em f$.

Deve notar-se que, se fizermos parar o programa e depois arrancarmos de novo por meio de RUN, todas as variáveis serão apagadas e depois recriadas. Portanto, as equações que armazenarmos não são permanentes do mesmo modo que as oito equações de exemplo, pois estas oito equações são recarregadas em memória a partir das declarações DATA sempre que executamos o programa. Por este motivo, incluímos no *menu* a opção ''Gravar e verificar programa'', a qual grava em *cassette* o programa e todas as suas variáveis, incluindo f$. Quando é recarregado em memória a partir de *cassette*, o programa arranca de novo automaticamente na linha 1000 e imprime o *menu*, mantendo intactas todas as variáveis e todas as equações em f$.

As curvas de exemplo

É surpreendentemente difícil predizer a forma de uma curva apenas com base no par de equações que a define. O círculo é o elemento básico de construção de todas as curvas fechadas, elipses e espirais e o par de equações para definir um círculo de raio 1 é o seguinte:

```
x = cos n
y = sin n
```

Nota: na notação portuguesa, a função ''seno'' é representada por ''sen'', sendo ''sin'' (de ''sinus'') a representação na notação inglesa, tal como o *Spectrum* e a maioria das calculadoras utilizam.

Quando este par de equações é representado graficamente por meio do programa ''Espiromania'', o resultado apresenta-se como um pequenino círculo no centro da área do *écran*, nas coordenadas x = 127 e y = 70. Do mesmo modo, um círculo de raio 60 terá as seguintes equações:

```
x = 60*cos n
y = 60*sin n
```

Os valores mínimo e máximo que ambas as funções sin e cos podem tomar são −1 e +1 e, portanto, o maior círculo que pode ser traçado com centro em x = 127, y = 70, sem sair da área do *écran*, é dado por:

```
x = 70*cos n
y = 70*sin n
```

Se desejarmos um círculo maior, até um raio máximo de 87, o centro do círculo deverá ser deslocado na direcção y para y = 87, o que se consegue alterando o valor de cy na linha 1730 do programa. O círculo seria então desenhado invadindo a área do *écran* reservada para apresentar o título e as equações.

Pode-se considerar uma elipse como um círculo que tenha sido «esticado» numa dada direcção. Assim, por exemplo, a maior elipse que pode ser acomodada dentro do *écran*, centrada em x = 127, y = 70, é definida pelas equações:

```
x = 127*cos n
y = 70*sin n
```

Para rodar uma elipse um certo ângulo, é necessário combinar as funções sin e cos nas equações de x e de y como, por exemplo, em:

```
x = 30*cos n + 15*sin n
y = 30*sin n + 15*cos n
```

Podemos obter efeitos interessantes combinando dois círculos entre si, como na segunda curva de exemplo, apresentada na figura 25. As equações são as seguintes:

$$x = 25*\cos(12*n) + 50*\cos n$$
$$y = 15*\sin(12*n) + 30*\sin n$$

```
                Spiromania

25*COS (12*n)+50*COS n
15*SIN (12*n)+30*SIN n
```

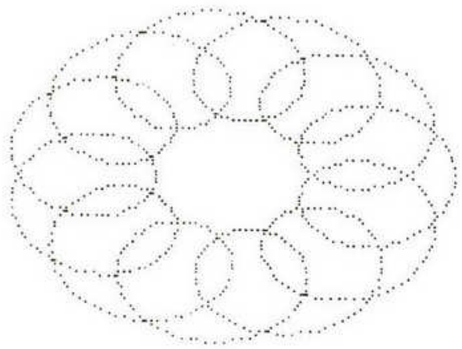


Figura 25. *Uma grande elipse com doze elipses mais pequenas sobrepostas (ver texto).*

A forma básica da figura é uma elipse com 50 unidades de comprimento e 30 unidades de altura, definida pelo segundo termo de cada uma das equações. Nesta figura estão sobrepostas doze elipses mais pequenas, cada uma com 25 unidades de comprimento e 15 unidades de altura, definidas pelo primeiro termo de cada equação. As figuras 26 e 27 apresentam outras curvas.

Podemos obter uma ideia das dimensões totais de uma figura adicionando os factores de multiplicação (coeficientes) das funções sin e cos. Desde que a soma desses coeficientes seja inferior a 127 na direcção x e menor do que 70 na direcção y, o desenho não sairá dos limites do *écran*.

Neste programa podemos utilizar outras funções em vez das funções sin e cos, ou mesmo em combinação com estas. Contudo, para evitar saídas do *écran* e obter curvas fechadas (isto é, curvas que, ao serem traçadas, voltam ao ponto de início e

repetem o caminho anteriormente percorrido), as funções devem ter uma cota superior e uma cota inferior finitas e devem assumir os mesmos valores ciclicamente. A função sin do exemplo nunca excede +1, nunca é inferior a −1 e sin (n+2π) = sin n, etc.

```
                Spiromania

70*COS (n/8)*COS n
50*SIN (n/8)*SIN n
```

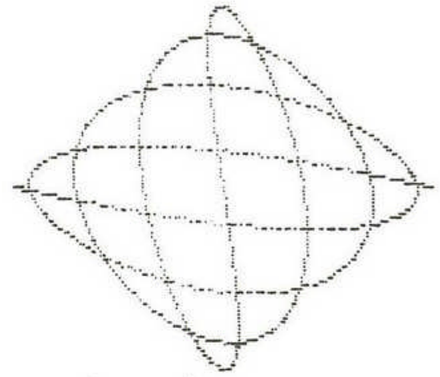


Figura 26. *Figura produzida multiplicando entre si funções sin e cos.*

```
                Spiromania

15*COS (10*n)+80*COS n+5*COS (n/
4)*COS (n/4)
15*SIN (10*n)+45*SIN n+5*SIN (n/
4)*SIN (n/4)
```

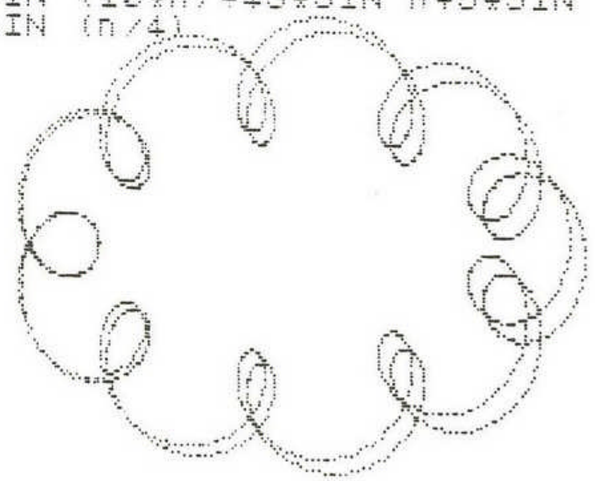


Figura 27. *Uma grande elipse com elipses mais pequenas sobrepostas e «perturbada» por um pequeno termo cos*cos e sin*sin.*

Programa 17. *«Spiromania».*

```
  50 REM "SPIROMANIA"
 100 BORDER 1: PAPER 1: INK 7
 110 LET olf=0: LET lf=0: LET tl
=1000
 120 DIM f$(tl)
 200 DATA 8
 210 DATA "50*COS n+50*COS (n/4)
","30*SIN n+30*SIN (n/4)","25*CO
S (12*n)+50*COS n","15*SIN (12*n
)+30*SIN n"
 220 DATA "50*COS n+50*SIN (n/6)
","30*SIN n+30*COS (n/6)"
 230 DATA "20*COS n+20*COS (n/2)
+20*COS (n/4)","15*SIN n+15*SIN
(n/2)+15*SIN (n/4)"
 240 DATA "70*COS (n/8)*COS n","
50*SIN (n/8)*SIN n"
 250 DATA "50*COS (n/8)*COS n+50
*COS (n/4)","35*SIN (n/8)*SIN n+
35*SIN (n/4)"
 260 DATA "40*SIN n+20*SIN (4*n)
+10*SIN (16*n)+5*SIN (32*n)","32
*COS n+8*COS (4*n)+4*COS (16*n)+
2*COS (32*n)"
 270 DATA "15*COS (10*n)+80*COS
n+5*COS (n/4)","15*SIN (10*n)+45
*SIN n+5*SIN (n/4)"
 300 READ nf
 310 FOR i=1 TO nf
 320 READ x$,y$
 330 IF LEN x$>255 OR LEN y$>255
 THEN BEEP .2,24: PRINT "A funca
o na declaracao DATA tem mais de
 255 caracteres": STOP
 340 LET lf=lf+LEN x$+LEN y$+2
 350 IF lf>=1000 THEN BEEP .2,24
: PRINT ,,"Nao ha'espaco suficie
nte p/arma-zenar as funcoes - in
cremente  tl na linha 20": STOP
 360 LET f$(olf+1 TO lf)=CHR$ LE
N x$+x$+CHR$ LEN y$+y$
 370 LET olf=lf
 380 NEXT i
 390 LET x$="": LET y$=""
1000 LET t=30: CLS : PRINT TAB 1
1;"Spiromania"
1010 PRINT ,,,,TAB 14;"Menu"
1020 PRINT ,,"Experimentar uma f
uncao";TAB t;"E","Desenhar uma c
urva";TAB t;"D","Memorizar uma f
```



```
uncao";TAB t;"M","Gravar e verif
icar o prog.";TAB t;"G","Parar o
 programa";TAB t;"P"
1030 INPUT z$: LET z$=CHR$ (CODE
 z$-32*(z$>"f"))
1040 GO TO 1200+300*(z$="E")+800
*(z$="D")+1300*(z$="M")+1800*(z$
="G")+2300*(z$="P")
1200 INPUT "Prima ENTER p/ volta
r ao menu";z$
1210 GO TO 1000
1500 PRINT ,,TAB 5;"Experimentar
 uma funcao"
1510 INPUT "Introduza a equacao
em x ";x$
1520 IF LEN x$=0 OR LEN x$>255 T
HEN BEEP .2,24: PRINT ,,"O compr
imento da equacao tem de ter de
1 a 255 caracteres.       Escreva
 de novo a equacao.": GO TO 1510
1530 PRINT ,,,,"A equacao em x e
' :"'';x$
1550 INPUT "Introduza a equacao
em y ";y$
1560 IF LEN y$=0 OR LEN y$>255 T
HEN BEEP .2,24: PRINT ,,"O compr
imento da equacao tem de ter de
1 a 255 caracteres.       Escreva
 de novo a equacao.": GO TO 1550
1570 PRINT ,,"A equacao em y e'
:"'';y$
1580 GO SUB 1600
1590 GO TO 1200
1600 INPUT "Deseja copia impress
a (S/N)?";z$
1610 LET z$=CHR$ (CODE z$-32*(z$
>"f"))
1700 CLS : PRINT TAB 11;"Spiroma
nia": PRINT #0;"** Prima uma tec
la para parar **"
1710 PRINT ,,x$: PRINT y$
1730 LET n=0: LET cx=127: LET cy
=70
1740 LET sx=cx+VAL x$: LET sy=cy
+VAL y$
1750 LET rx=VAL x$: LET ry=VAL y
$
1760 PLOT cx+rx,cy+ry
1770 LET n=n+.025
1780 IF INKEY$<>"" THEN GO TO 18
00
1790 GO TO 1750-550*(ABS (rx-sx)
```

```
<2 AND ABS (ry-sy)<2)
1800 IF z$="S" THEN COPY
1850 RETURN
2000 PRINT ,,"Desenhar uma funca
o"
2010 PRINT ,,"As funcoes disponi
veis sao :"
2020 LET f=0: LET n=1
2030 GO SUB 2400
2040 IF s>=lf THEN GO TO 2100
2050 PRINT ,,n;TAB 6;f$(s TO f);
TAB 6;
2060 GO SUB 2400
2070 PRINT f$(s TO f)
2080 LET n=n+1
2090 GO TO 2030
2100 INPUT "Introduza o numero d
a funcao: ";m
2110 IF m>n OR n<0 THEN BEEP .2,
24: GO TO 2100
2120 LET f=0: LET n=1
2130 GO SUB 2400
2140 IF n=m THEN LET x$=f$(s TO
f)
2150 GO SUB 2400
2160 IF n<>m THEN LET n=n+1: GO
TO 2130
2170 LET y$=f$(s TO f)
2180 GO SUB 1800
2190 LET x$=""
2200 LET y$=""
2210 GO TO 1200
2400 LET s=f+2
2410 LET f=s+CODE f$(s-1)-1
2420 RETURN
2500 PRINT ,,;TAB 6;"Memorizar u
ma funcao"
2510 IF LEN x$=0 OR LEN y$=0 THE
N BEEP .2,24: PRINT ,,"A funcao
nao esta' definida": GO TO 1200
2520 LET lf=lf+LEN x$+LEN y$+2
2530 IF lf>=1000 THEN BEEP .2,24
: PRINT ,,"Nao ha' espaco sufici
ente para  memorizar a funcao":
LET lf=lf-LEN x$-LEN y$-2: GO TO
 1200
2540 LET f$(olf+1 TO lf)=CHR$ LE
N x$+x$+CHR$ LEN y$+y$
2550 LET olf=lf
2560 PRINT ,,"Funcao em memoria"
2570 GO TO 1200
3000 PRINT ,,TAB 7;"Gravar Spiro
mania"
```



```
3010 SAVE "Spiromania" LINE 1000
3020 PRINT FLASH 1;AT 15,8;"PARE
 O GRAVADOR": INPUT "Quer verifi
car (S/N) ? ";z$
3030 IF z$<>"S" AND z$<>"s" THEN
 GO TO 1200
3040 PRINT AT 11,6;"Verificar Sp
iromania";AT 13,0;"- Rebobine a
cassette"''"- Coloque a ligacao
""EAR""",''"- Passe a gravacao"
3050 LET z$="G"
3060 VERIFY "Spiromania"
3070 PRINT ,,"Copia verificada"
3080 GO TO 1200
3500 PRINT ,,"O programa parou"
3510 STOP
```

TIRO AOS PATOS

A estrutura complexa do ficheiro de imagem do *Spectrum*, descrita no capítulo 24 do manual de programação BASIC do *ZX Spectrum*, torna bastante difícil manipular a imagem do *écran* a uma velocidade satisfatória utilizando um programa em BASIC. Este programa apresenta uma técnica destinada a alterar a imagem do *écran* a uma velocidade razoável através da manipulação do ficheiro de atributos e não do ficheiro de imagem. Uma segunda versão do programa, também aqui apresentada, emprega uma pequena rotina em código máquina para aumentar a velocidade da acção.

O programa simula as barracas de tiro aos patos que encontramos em muitas feiras. Quatro linhas de patos deslocam-se em parada para trás e para a frente na parte superior do *écran*. O jogador dispõe de uma única «espingarda», visível na parte de baixo do *écran*, que ele pode deslocar para a direita ou para a esquerda por meio das teclas 5 e 8 do teclado do *Spectrum*, servindo a tecla com o número 7 para a disparar. O objectivo do jogo consiste em abater todos os patos, utilizando para isso o menor número possível de «cartuchos» e de deslocações da «espingarda» no mínimo espaço de tempo.

O programa 18 é a versão do programa totalmente em BASIC. A primeira linha chama a sub-rotina na linha 7000 para definir

quatro caracteres gráficos definíveis pelo utilizador (UDG — User Defined Graphics), que irão representar a bala (B), a espingarda (G), um pato a deslocar-se para a direita (D) e um pato a deslocar-se para a esquerda (E). O jogador escolhe seguidamente as cores que deseja para o fundo (p) e para a «tinta» (i), ou seja, para os «patos», e o programa apresenta então a área de jogo, ilustrada na figura 28.

Figura 28. *O aspecto gráfico de «Tiro aos patos».*

Repare-se que as declarações que imprimem os patos (linhas 200 a 290) são bastante invulgares. Cada uma das oito linhas impressas consiste numa sequência repetitiva de um pato na cor da «tinta», i, seguido de três patos na cor do fundo, p. Os patos impressos na cor do fundo ficam invisíveis contra o fundo da mesma cor e, portanto, a imagem parecerá conter apenas um quarto dos patos que na realidade estão presentes.

O jogo em si é comandado pelo ciclo das linhas 2000 a 2110. O ciclo chama uma sub-rotina situada a partir da linha 2200, rotina essa cuja função consiste em fazer alternar 32 *bytes,* deslocando-os ciclicamente para a direita ou para a esquerda no ficheiro de atributos. O sentido da deslocação cíclica é determinado pelo valor de d. O efeito desta alternância é mover o atributo «tinta» um carácter de cada vez para a direita ou para a esquerda, ao longo de uma linha de 32 caracteres, de modo a que os patos que se encontravam anteriormente visíveis porque o seu *byte* de atributo possuía o valor de «tinta» i se tornem invisíveis.

De modo semelhante, um dos três conjuntos de patos que se encontravam anteriormente invisíveis por o seu *byte* de atributos possuir o valor de «tinta» igual ao do fundo, p, torna-se visível. Desta forma consegue-se criar a ilusão de movimento.

A sub-rotina da linha 2200 também chama mais três sub-rotinas. A função da primeira, com início na linha 3000, é apagar a bala do *écran* antes de mover os patos. A segunda rotina, que começa na linha 3100, faz a bala aparecer de novo no *écran* depois de os patos terem sido movidos e desloca-a para cima. Se um pato ocupar a posição de carácter na qual a bala vai entrar, tanto esta como o pato são apagados imprimindo novamente o pato correspondente, mas desta vez com o atributo «tinta» (INK) fixado na cor do fundo. A sub-rotina da linha 4000 dispara a «espingarda».

A segunda versão do programa, o programa 18-A, utiliza uma rotina em código máquina para movimentar o ficheiro de atributos. A rotina é carregada em memória a partir da declaração DATA da linha 20, numa área situada entre RAMTOP e a área reservada aos caracteres gráficos definidos pelo utilizador, sendo o carregamento efectuado pelo ciclo das linhas 30 a 60. A rotina em código máquina encontra-se listada na tabela 9.

Tabela 9. *Esta rotina alterna as linhas 0, 2, 4 e 6 do ficheiro de atributos.*

Hexadecimal		*Mnemónica Assembler*
06 02		LD B,2
21 00 58		LD HL, 22528
7E	NEXT	LD A, (HL)
C5		PUSH BC
54		LD D,H
5D		LD E, L
23		INC HL
01 1F 00		LD BC, 31
ED B0		LDIR
2B		DEC HL
77		LD (HL), A
01 40 00		LD BC, 64

09	ADD AL, BC
7E	LD A, (HL)
54	LD D, H
5D	LD E, L
2B	DEC HL
01 1F 00	LD BC, 31
ED B8	LDDR
23	INC HL
77	LD (HL), A
01 40 00	LD BC, 64
09	ADD HL, BC
C1	POP BC
10 DE	DJNZ, −34 NEXT
C9	RET

Programa 18. *A versão totalmente em BASIC de «Tiro aos patos».*

```
  5 REM "TIRO AOS PATOS"
 10 REM UDG a=B b=G c=D d=E
 20 GO SUB 7000
100 PRINT "Introduza a cor do f
undo:": INPUT p: IF p<0 OR p>7 T
HEN BEEP 1,12: GO TO 100
110 PRINT "Introduza a cor dos
patos:": INPUT i: IF i<0 OR i>7
OR i=p THEN BEEP 1,12: GO TO 110
120 BORDER p: PAPER p: INK i: C
LS
130 LET real=8*p+i
140 LET l=20: LET cg=16: LET h=
0: LET cy=-50
150 PRINT AT l,cg;"a"
160 PRINT AT 0,0;
200 FOR j=1 TO 2
210 FOR k=1 TO 8
220 PRINT INK i;"b"; INK p;"bbb
";
230 NEXT k
240 PRINT INK p,,
250 FOR k=1 TO 8
260 PRINT INK p;"c"; INK i;"c";
INK p;"cc";
270 NEXT k
280 PRINT INK p,,
```



```
290 NEXT j
300 PRINT INK i,,,,,,,,,,,,,,,,
,,,,,,,,
2000 LET s=22528
2010 FOR j=1 TO 2
2020 LET d=1
2030 GO SUB 2200
2040 PAUSE 1: IF h>=32 THEN PRIN
T AT 5,2; FLASH 1;"FIM DO JOGO.
PRIMA UMA TECLA": PAUSE 0: RUN
2050 LET s=s+35
2060 LET d=-1
2070 GO SUB 2200
2080 PAUSE 1: IF h>=32 THEN PRIN
T AT 5,2; FLASH 1;"FIM DO JOGO.
PRIMA UMA TECLA": PAUSE 0: RUN
2090 LET s=s+33
2100 NEXT j
2110 GO TO 2000
2200 LET a=PEEK s
2210 FOR i=0 TO 30*d STEP d
2215 GO SUB 3000
2220 POKE s+i,PEEK (s+i+d)
2230 GO SUB 3100
2240 GO SUB 4000
2250 NEXT i
2260 POKE s+d*31,a
2270 LET cy=cy+50
2280 PRINT AT 21,0;"Abatidos = "
;h;TAB 14;"Pontuacao = ";h*100-c
y,
2290 RETURN
3000 IF l=20 THEN RETURN
3010 PRINT INK p; PAPER p;AT l,c
;" "
3020 IF l=0 OR l=4 THEN PRINT AT
 l,c; INK p; PAPER p;"b"
3030 IF l=2 OR l=6 THEN PRINT AT
 l,c; INK p; PAPER p;"c"
3040 LET l=l-2: RETURN
3100 IF l<0 THEN LET l=20: RETUR
N
3110 IF l=20 THEN RETURN
3120 IF l>7 THEN PRINT AT l,c;"a
": RETURN
3130 IF ATTR (l,c)=real THEN PRI
NT AT l,c; INK p; PAPER p;" ": L
ET h=h+1: LET l=20: RETURN
3140 PRINT AT l,c;"a": RETURN
3150 RETURN
4000 LET k=PEEK 23560: POKE 2356
0,0
```

```
4010 GO TO 4100+100*(k=55)+200*(
k=53)+300*(k=56)
4100 RETURN
4200 IF l<>20 THEN RETURN
4210 LET l=l-2: LET c=cg: LET cy
=cy+30
4220 PRINT AT l,c;"▝▘"
4230 RETURN
4300 IF cg<>0 THEN LET cg=cg-1
4310 PRINT AT 20,cg+1;" "
4320 PRINT AT 20,cg;"▛▜"
4330 LET cy=cy+5
4340 RETURN
4400 IF cg<>31 THEN LET cg=cg+1
4410 PRINT AT 20,cg-1;" "
4420 PRINT AT 20,cg;"▛▜"
4430 LET cy=cy+5
4440 RETURN
7000 POKE USR "B",0
7001 POKE USR "B"+1,0
7002 POKE USR "B"+2,16
7003 POKE USR "B"+3,16
7004 POKE USR "B"+4,16
7005 POKE USR "B"+5,16
7006 POKE USR "B"+6,0
7007 POKE USR "B"+7,0
7008 POKE USR "G",0
7009 POKE USR "G"+1,24
7010 POKE USR "G"+2,24
7011 POKE USR "G"+3,24
7012 POKE USR "G"+4,24
7013 POKE USR "G"+5,24
7014 POKE USR "G"+6,24
7015 POKE USR "G"+7,0
7016 POKE USR "D",0
7017 POKE USR "D"+1,12
7018 POKE USR "D"+2,14
7019 POKE USR "D"+3,4
7020 POKE USR "D"+4,60
7021 POKE USR "D"+5,124
7022 POKE USR "D"+6,60
7023 POKE USR "D"+7,0
7024 POKE USR "E",0
7025 POKE USR "E"+1,48
7026 POKE USR "E"+2,112
7027 POKE USR "E"+3,32
7028 POKE USR "E"+4,60
7029 POKE USR "E"+5,62
7030 POKE USR "E"+6,60
7031 POKE USR "E"+7,0
7032 RETURN
```


Programa 18-A. *Esta versão de «Tiro aos patos» emprega uma rotina em código máquina para manipular o ficheiro de atributos.*

```
   5 REM "TIRO AOS PATOS II"
   6 REM UDG ▝=B ▛=G ▟=D ▙=E
  10 CLEAR 35559
  20 DATA 6,2,33,0,88,126,197,84
,93,35,1,31,0,237,176,43,119,1,5
4,0,9,126,84,93,43,1,31,0,237,16
4,35,119,1,64,0,9,193,16,222,201
  30 FOR i=0 TO 39
  40 READ j
  50 POKE 32560+i,j
  60 NEXT i
  70 GO SUB 7000
  80 LET r=0
 100 PRINT "Introduza a cor do f
undo:": INPUT p: IF p<0 OR p>7 T
HEN BEEP 1,12: GO TO 100
 110 PRINT "Introduza a cor dos
patos:": INPUT i: IF i<0 OR i>7
OR i=p THEN BEEP 1,12: GO TO 110
 120 BORDER p: PAPER p: INK i: C
LS
 130 LET real=8*p+i
 140 LET l=20: LET cg=16: LET h=
0: LET cy=0
 150 PRINT AT l,cg;"▛▜"
 160 PRINT AT 0,0;
 200 FOR j=1 TO 2
 210 FOR k=1 TO 8
 220 PRINT INK i;"▙"; INK p;"▙▙▙
";
 230 NEXT k
 240 PRINT INK p,,
 250 FOR k=1 TO 8
 260 PRINT INK p;"▟"; INK i;"▟";
 INK p;"▟▟";
 270 NEXT k
 280 PRINT INK p,,
 290 NEXT j
 300 PRINT INK i,,,,,,,,,,,,,,,,
,,,,,,,,
2000 LET s=22528
2010 FOR j=1 TO 2
2020 LET d=1
2030 GO SUB 2200
2040 IF h>=32 THEN PRINT AT 5,2;
 FLASH 1;"FIM DO JOGO. PRIMA UMA
 TECLA": PAUSE 0: PAUSE 0: RUN
2050 LET s=s+95
```

```
2060 LET d=-1
2070 GO SUB 2200
2080 IF h>=32 THEN PRINT AT 5,2;
 FLASH 1;"FIM DO JOGO. PRIMA UMA
 TECLA": PAUSE 0: PAUSE 0: RUN
2090 LET s=s+33
2100 NEXT j
2110 GO TO 2000
2215 GO SUB 3000
2220 LET r=r+1-8*(r=7): IF r=0 T
HEN RANDOMIZE USR 32560: LET cy=
cy+10
2230 GO SUB 3100
2240 GO SUB 4000
2280 PRINT AT 21,0;"Abatidos = "
;h;TAB 14;"Pontuacao = ";h*100-c
y,
2290 RETURN
3000 IF l=20 THEN RETURN
3010 PRINT INK p; PAPER p;AT l,c
;" "
3020 IF l=0 OR l=4 THEN PRINT AT
 l,c; INK p; PAPER p;"b"
3030 IF l=2 OR l=6 THEN PRINT AT
 l,c; INK p; PAPER p;"a"
3040 LET l=l-2: RETURN
3100 IF l<0 THEN LET l=20: RETUR
N
3110 IF l=20 THEN RETURN
3120 IF l>7 THEN PRINT AT l,c;"
": RETURN
3130 IF (l=0 OR l=4) AND ATTR (l
,c)=real THEN PRINT AT l,c; INK
p; PAPER p;"b": LET h=h+1: LET l
=20: RETURN
3135 IF (l=2 OR l=6) AND ATTR (l
,c)=real THEN PRINT AT l,c; INK
p; PAPER p;"a": LET h=h+1: LET l
=20: RETURN
3140 PRINT AT l,c;"": RETURN
3150 RETURN
4000 LET k=PEEK 23560: POKE 2356
0,0
4010 GO TO 4100+100*(k=55)+200*(
k=53)+300*(k=56)
4100 RETURN
4200 IF l<>20 THEN RETURN
4210 LET l=l-2: LET c=cg: LET cy
=cy+30
4220 PRINT AT l,c;""
4230 RETURN
4300 IF cg<>0 THEN LET cg=cg-1
```



```
4310 PRINT AT 20,cg+1;" "
4320 PRINT AT 20,cg;""
4330 LET cy=cy+5
4340 RETURN
4400 IF cg<>31 THEN LET cg=cg+1
4410 PRINT AT 20,cg-1;" "
4420 PRINT AT 20,cg;""
4430 LET cy=cy+5
4440 RETURN
7000 POKE USR "B",0
7001 POKE USR "B"+1,0
7002 POKE USR "B"+2,16
7003 POKE USR "B"+3,16
7004 POKE USR "B"+4,16
7005 POKE USR "B"+5,16
7006 POKE USR "B"+6,0
7007 POKE USR "B"+7,0
7008 POKE USR "G",0
7009 POKE USR "G"+1,24
7010 POKE USR "G"+2,24
7011 POKE USR "G"+3,24
7012 POKE USR "G"+4,24
7013 POKE USR "G"+5,24
7014 POKE USR "G"+6,24
7015 POKE USR "G"+7,0
7016 POKE USR "D",0
7017 POKE USR "D"+1,12
7018 POKE USR "D"+2,14
7019 POKE USR "D"+3,4
7020 POKE USR "D"+4,60
7021 POKE USR "D"+5,124
7022 POKE USR "D"+6,60
7023 POKE USR "D"+8,0
7024 POKE USR "E",0
7025 POKE USR "E"+1,48
7026 POKE USR "E"+2,112
7027 POKE USR "E"+3,32
7028 POKE USR "E"+4,60
7029 POKE USR "E"+5,62
7030 POKE USR "E"+6,60
7031 POKE USR "E"+7,0
7032 RETURN
```

DIGITIZADOR

Com este programa podemos gerar no *écran* a imagem de um mapa, diagrama, anúncio ou qualquer outra figura. Tem dispositivos que nos permitem aumentar ou diminuir a escala dos valores que introduzimos, desenhar uma imagem constituída apenas por linhas (*line image* ou imagem de linhas) e colorir a área que se estende verticalmente entre duas partes da imagem de linhas. Podemos retocar a imagem *pixel* por *pixel*, e alterar os atributos de cada célula de carácter directamente a partir do programa. Também podemos gravar a imagem em *cassette*. Todas as instruções são transmitidas ao programa sem que a imagem deixe de se manter no *écran*.

A figura 29 apresenta um mapa de parte da costa ocidental de Portugal, à latitude de Lisboa, tal como é produzido pelo programa. As coordenadas da linha de costa foram tiradas de um mapa à escala de 1/350 000, fazendo corresponder cada *pixel* a um milímetro do mapa.

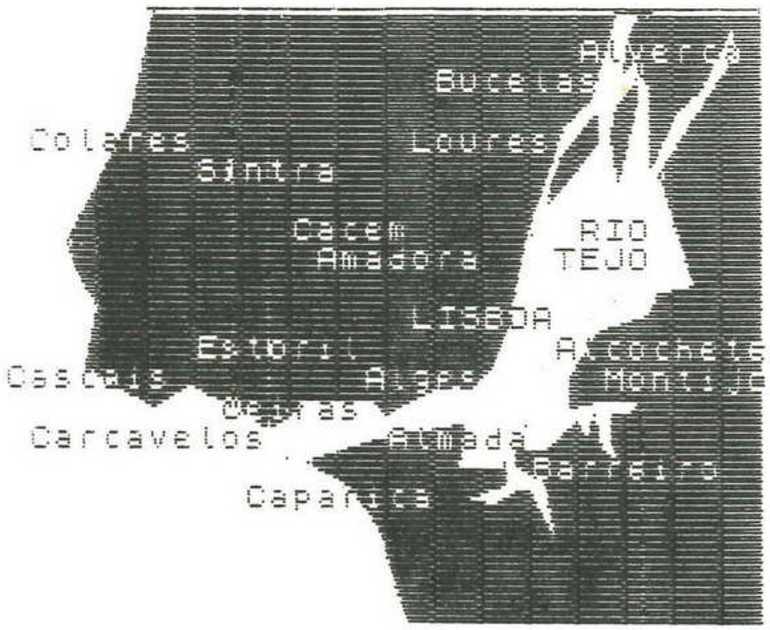


Figura 29. *Um mapa desenhado por meio do «Digitizador».*

Forma de usar o «Digitizador»

Quando é executado, o programa solicita as coordenadas X e Y a atribuir ao canto inferior esquerdo do diagrama e a coordenada X a atribuir ao canto superior direito do diagrama. Seguidamente, calcula um factor de escala apropriado e um valor máximo para a coordenada Y do canto superior direito, a partir das duas coordenadas X e da coordenada Y introduzidas por nós, perguntando-nos seguidamente se achamos satisfatórios os valores das coordenadas entretanto apresentados. O factor de escala é aplicado a todos os valores subsequentemente introduzidos e rejeitado qualquer valor que se situe fora dos limites de X e Y definidos do modo indicado.

O programa regula a escala de cada diagrama e posiciona-o no *écran* de forma a que cubra uma área que pode ir do *pixel* 1 ao *pixel* 254, inclusive, na direcção X, e do *pixel* 9 ao *pixel* 174 na direcção Y. Isto deixa uma margem com um *pixel* de largura à esquerda, à direita e por cima do diagrama, de forma a evitar que eventuais erros de arredondamento em cálculos subsequentes afectos a PLOT e DRAW façam o diagrama ultrapassar o limite do *écran* e parar o programa com uma mensagem de erro. Fica uma linha vazia na parte de baixo do *écran*, de modo a que o programa e o operador possam comunicar entre si.

O programa pede-nos também que seleccionemos um número conveniente que se situe fora dos limites admitidos para os valores de X. Este número fica subsequentemente disponível para o usarmos como um «número de retorno» para fazer o programa voltar ao *menu* principal. Também se nos pede que seleccionemos valores iniciais para as cores da margem, do fundo e da tinta.

Ao contrário da maioria dos programas deste volume, este programa não imprime um *menu* principal porque, se o fizesse, iria sobrepô-lo à imagem apresentada nesse momento no *écran* e, portanto, estragava-a. Assim, toda a comunicação entre nós e o programa é efectuada por meio da linha de INPUT, na parte de baixo do *écran*. Consequentemente, as mensagens emitidas pelo programa são bastante lacónicas.

A tabela 10 apresenta a listagem das instruções que podemos introduzir sob a forma de uma só letra de comando, letra essa apresentada para cada instrução. Introduzida a letra apropriada, o programa salta da linha 1030 para a rotina correspondente.

Tabela 10. *As instruções que podemos introduzir.*

Instrução	*Letra de Comando*	*Comentário*
Desenhar	D	Utilizada para introduzir pares de coordenadas destinadas a criar uma imagem de linhas.
Cor	C	Iniciar a coloração da totalidade ou só de parte da área de *écran* situada dentro ou fora da imagem de linhas.
Tinta e Fundo	T	Definir as cores da tinta e do fundo a usar subsequentemente.
Atributo	A	Definir as cores da tinta e do fundo de células de carácter seleccionadas.
Palavras	P	Colocar uma palavra numa dada posição do *écran*.
Retocar	R	Imprimir um ponto numa dada posição do *écran*.
Gravar (Save)	S	Gravar em *cassette* os dados que definem a imagem apresentada no *écran*.
Carregar (Load)	L	Carregar em memória os dados definidores de uma imagem.
Imprimir	I	Copiar através da impressora a imagem apresentada no *écran*.
Parar (Stop)	S	Parar o programa.

Desenhar Introduzir D

É com esta rotina que são geradas as imagens de linhas que definem o diagrama. O programa verifica se o par de coordenadas que introduzimos se situa dentro dos limites permitidos, afecta-os de um factor de escala e desenha a linha correspondente no *écran*. Os pares de coordenadas também vão sendo impressos pela impressora, se esta estiver acopulada. Podemos continuar a introduzir pares de números até introduzirmos o número de retorno escolhido para obter o *menu* principal (que, como se disse, não é impresso). A função DRAW OVER 1 é utilizada nesta opção de modo a que, se tiver sido cometido um erro, a repetição da instrução (imprimir a mesma linha, mas em sentido contrário) apague a linha indesejada.

Cor Introduzir C

Esta opção imprime *pixels,* na cor de tinta que esteja a ser utilizada, na totalidade ou só na parte do *écran* situada entre as linhas que definem a imagem. Definimos uma «caixa» dentro da qual se deve efectuar a coloração do diagrama e, depois de verificar que a caixa não ultrapassa os limites do *écran,* o programa traça (PLOTs) a dita caixa no *écran*. A coloração é iniciada no canto inferior esquerdo da caixa e prossegue para a direita, traçando sucessivas colunas verticais de *pixels*.

O programa pergunta-nos se a «função de coloração» deve ser «ligada» ou «desligada», no início da coloração e sempre que a posição de traçagem (PLOT *position*) atinja uma imagem pre-existente.

A nossa atenção é chamada para a posição de traçagem corrente através da utilização da função FLASH 1, que faz «piscar» a célula de carácter que contém a posição de traçagem.

Tinta e fundo Introduzir T

Esta opção altera os valores da tinta e do fundo que estiverem a ser utilizados (valores correntes).

Atributos, palavras e retoque Introduzir A, P ou R

Pela opção «Atributos» podemos atribuir valores especificados por nós aos atributos INK (tinta) e PAPER (fundo) de uma célula de carácter que tivermos escolhido. A opção «Palavras» coloca uma legenda numa posição ou célula de carácter também deter-

minada por nós, com os atributos INK e PAPER então em vigor (ou seja, nas cores de tinta e de fundo que estejam a ser utilizadas na altura). Por fim, a opção «Retocar» imprime, por meio de PLOT, um ponto *(pixel)* na posição definida por um dado par de coordenadas X, Y. Em qualquer das três opções é utilizada a função OVER 1, de modo a que a repetição da instrução provoque o apagamento ou anulação do resultado anterior dessa mesma instrução, corrigindo assim erros eventuais. Com o número de retorno obtém-se o *menu*.

Programa 19. *«Digitizador».*

```
 100 REM "DIGITIZADOR"
 120 PRINT TAB 10;"Digitizador"'
'"Introduza coordenadas do canto
  inferior esquerdo:"
 130 PRINT ,,"Canto inf. esquerd
o = ";
 140 INPUT xl
 150 PRINT xl;",";
 160 INPUT yl.
 170 PRINT yl
 180 PRINT ,,"Introduza a coorde
nada X do can-to superior direit
o:"
 190 INPUT xu: IF xu<=xl THEN BE
EP .2,24: GO TO 190
 200 LET yu=INT ((xu-xl)*165/253
-yl)
 210 PRINT ,,"Canto sup. direito
  = ";xu;",";yu
 220 PRINT ,,"Esta' correcto (S
ou N) ? "
 230 INPUT z$: IF z$="N" OR z$="
n" THEN RUN
 240 PRINT ,,"introduza um numer
o adequado quenao esteja entre "
;xl;" e ";xu;".": PRINT "Este nu
mero sera' usado para fa-zer o p
rograma voltar ao MENU."
 250 INPUT rx: IF rx>=xl AND rx<
=xu THEN BEEP .2,24: GO TO 250
 310 PRINT "Margem = ";
 320 INPUT m: IF m<0 OR m>7 THEN
 BEEP .2,24: GO TO 320
 330 PRINT m
 340 PRINT "Fundo  = ";
 350 INPUT f: IF f<0 OR f>7 THEN
 BEEP .2,24: GO TO 350
 360 PRINT f;TAB 0;"Tinta  = ";
 380 INPUT t: IF t<0 OR t>7 THEN
 BEEP .2,24: GO TO 380
 390 PRINT t
 400 BORDER m: PAPER f: INK t
 990 CLS
1010 INPUT "MENU - introduza ins
trucao: ";z$
1020 LET y$=CHR$ (CODE z$-32*(z$
>"£"))
1030 GO TO 1100+4400*(y$="A")+49
00*(y$="P")+3900*(y$="R")+3400*(
y$="L")+400*(y$="C")+900*(y$="T"
)+1400*(y$="D")+1900*(y$="G")+24
00*(y$="S")+2900*(y$="I")
1100 GO TO 1000
1500 INPUT "Cor - canto inf. e.
";xlb;",";ylb
1510 IF xlb<xl OR xlb>xu OR ylb<
yl OR ylb>yu THEN BEEP .2,24: GO
 TO 1500
1520 INPUT "Cor - canto sup. d.
";xub;",";yub
1530 IF xub<xlb+2 OR xub>xu OR y
ub<ylb+2 OR yub>yu THEN BEEP .2,
24: GO TO 1520
1540 LET xlb=xlb*253/(xu-xl)+1
1550 LET xub=xub*253/(xu-xl)+1
1560 LET ylb=ylb*165/(yu-yl)+9
1570 LET yub=yub*165/(yu-yl)+9
1580 GO SUB 9000
1590 INPUT "Cor - ""caixa"" OK (
S ou N) ? ";z$
1600 IF z$<>"N" AND z$<>"n" THEN
 GO TO 1630
1610 GO SUB 9000
1620 GO TO 1500
1650 INPUT "Cor - ligada/deslig.
 (L/D)?";z$
1660 IF z$<>"l" AND z$<>"L" AND
z$<>"d" AND z$<>"D" THEN BEEP .2
,24: GO TO 1650
1700 FOR j=xlb+1 TO xub-1
1710 LET y=(z$="l"): LET op=0
1720 FOR k=ylb+1 TO yub
1730 LET q=POINT (j,k)
1740 IF q=0 AND op=1 THEN GO SUB
 1800
1750 IF y=1 THEN PLOT INK t; PAP
ER f;j,k
```

```
1755 LET op=q:
1760 NEXT k
1770 NEXT j
1780 GO SUB 9000
1790 GO TO 1000
1805 PLOT INK t; FLASH 1; PAPER
f; OVER 1;j,k
1810 BEEP .2,24
1820 INPUT "Cor - ligada/deslig.
 (L/D)?";y$
1830 IF y$<>"l" AND y$<>"L" AND
y$<>"d" AND y$<>"D" THEN BEEP .2
,24: GO TO 1820
1840 LET y=(y$="l"): LET op=0
1850 PLOT INK t; FLASH 0; PAPER
f; OVER 1;j,k
1860 RETURN
2000 INPUT "Introduza nova cor d
o fundo: ";f
2010 IF f<0 OR f>7 THEN BEEP .2,
24: GO TO 2000
2020 INPUT "Introduza nova cor d
a tinta ";t
2030 IF t<0 OR t>7 THEN BEEP .2,
24: GO TO 2020
2040 GO TO 1000
2500 LPRINT : LPRINT "X  coorden
adas   Y"
2510 INPUT "Desenhar -int. orige
m ";ox;",";oy
2520 IF ox=rx THEN GO TO 1100
2530 IF ox<xl OR ox>xu OR oy<yl
OR oy>yu THEN BEEP .2,24: GO TO
2510
2535 LPRINT ox,oy
2540 LET ox=ox*253/(xu-xl)+1
2550 LET oy=oy*165/(yu-yl)+9
2560 PLOT OVER 1; INK t;ox,oy
2600 INPUT "Desenhar - int. coor
d ";x;",";y
2610 IF x=rx THEN GO TO 1100
2620 IF x<xl OR x>xu OR y<yl OR
y>yu THEN BEEP .2,24: GO TO 2600
2625 LPRINT x,y
2630 LET x=x*253/(xu-xl)+1
2640 LET y=y*165/(yu-yl)+9
2650 LET px=x-ox
2660 LET py=y-oy
2670 DRAW OVER 1; INK t;px,py
2680 LET ox=x
2690 LET oy=y
2700 GO TO 2600
```



```
3000 INPUT "Gravar - int. nome "
;z$
3010 IF LEN z$=0 OR LEN z$>10 TH
EN BEEP .2,24: GO TO 3000
3020 SAVE z$SCREEN$
3030 GO TO 1000
3500 STOP
4000 COPY
4010 GO TO 1000
4500 INPUT "Carregar - int. nome
 ";z$
4510 IF LEN z$>10 THEN BEEP .2,2
4: GO TO 4500
4520 LOAD z$SCREEN$
4530 GO TO 1000
5000 INPUT "Retocar - int. coord
. ";x;",";y
5010 IF x=rx THEN GO TO 1000
5020 LET x=x*253/(xu-xl)+1
5030 LET y=y*165/(yu-yl)+9
5040 PLOT OVER 1; INK t; PAPER f
;x,y
5050 GO TO 5000
5500 INPUT "Atri. - int. coord.
AT ";r;",";c
5510 IF r=rx THEN GO TO 1000
5520 IF r>21 OR r<0 OR c>31 OR c
<0 THEN BEEP .2,24: GO TO 5500
5530 PRINT INK t; PAPER f; OVER
1;AT r,c;" ";
5540 GO TO 5500
6000 INPUT "Palav. - int. coord.
 AT ";r;" ";c
6010 IF r=rx THEN GO TO 1000
6020 IF r>21 OR r<0 OR c>31 OR c
<0 THEN BEEP .2,24: GO TO 6000
6030 INPUT "Palavras ?";z$
6040 PRINT INK t; PAPER f; OVER
1;AT r,c;z$
6050 INPUT "Esta' correcto (S ou
 N) ? ";y$
6060 IF y$="N" OR y$="n" THEN PR
INT INK t; PAPER f; OVER 1;AT r,
c;z$
6070 GO TO 6000
9000 PLOT INK t; PAPER f; OVER 1
;xlb,ylb
9010 DRAW INK t; PAPER f; OVER 1
;xub-xlb,0
9020 DRAW INK t; PAPER f; OVER 1
;0,yub-ylb
9030 DRAW INK t; PAPER f; OVER 1
```

```
;xlb-xub,0
9040 DRAW INK t; PAPER f; OVER 1
;0,ylb-yub+1
9050 RETURN
```

CALEIDOSCÓPIO

Para completar esta colecção dos 20 melhores programas para o *ZX Spectrum,* apresentamos o programa «Caleidoscópio». É um programa simples, mas que certamente hipnotizará o utilizador desprevenido.

O programa funciona introduzindo por meio de POKE uma sequência flutuante de números no ficheiro de atributos e reflectindo os valores em oito quadrantes. A dita sequência é gerada pela equação da linha 70. Praticamente qualquer equação poderia ter sido utilizada, mas esta apresenta uma combinação satisfatória de estabilidade e variação. As linhas 80 a 150 introduzem por meio de POKE o valor de k, gerado pela equação mencionada, nos oito quadrantes, sendo cada quadrante um reflexo aproximado dos seus vizinhos.

Programa 20. *«Caleidoscópio».*

```
  5 REM "CALEIDOSCOPIO"
 10 LET d=0
 20 LET s=22928
 30 LET l=0
 40 FOR i=1 TO 12
 50 FOR j=0 TO l
 60 LET d=d+.03
 70 LET k=INT (127*SIN d)
 80 POKE s+j,k
 90 POKE s-j-1,k
100 POKE s+32-31*i+32*j,k
110 POKE s-31*i-32*j,k
120 POKE s-33*i+31+32*j,k
130 POKE s-33*i-1-32*j,k
140 POKE s-64*i+31-j,k
150 POKE s-54*i+32+j,k
160 NEXT j
170 LET l=l+1
180 LET s=s+32
190 NEXT i
200 GO TO 20
```