O Spectrum Funcional baseia-se numa colecção de programas sólidos e sofisticados, em áreas como armazenamento de dados, finança, cálculo, gráficos, administração doméstica e educação.

Cada programa é explicado em pormenor, linha após linha. E cada programa é construído a partir de sub-rotinas e módulos de aplicação geral que, uma vez compreendidos, podem formar a base de quaisquer outros programas que precise de escrever.

Técnicas de programação avançadas ressaltam das discussões explicativas de cada sub-rotina. Resulta daqui não só a melhoria das capacidades de programação do leitor como a obtenção de uma vasta gama de programas de aplicação prática que, de outro modo, apenas estariam disponíveis para os que estivessem preparados para comprar cassetes ou para escrever programas substanciais para si próprios.

O autor, David Lawrence, é um colaborador regular da publicação *Popular Computing Weekly*.

ARTE DE VIVER®
97

3309785

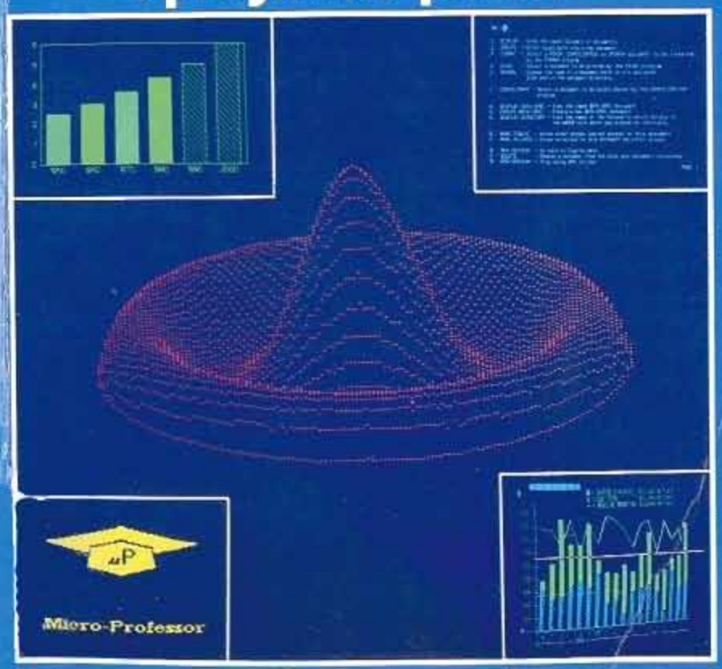
DAVID LAWRENCE
O SPECTRUM FUNCIONAL

ARTE DE VIVER
97

DAVID LAWRENCE

O SPECTRUM FUNCIONAL

uma biblioteca de sub-rotinas e programas práticos



PUBLICAÇÕES EUROPA-AMÉRICA

Título original: The Working Spectrum

Tradução de António José dos Santos Realinho

Capa: estúdios P. E. A.

First published in English 1983 by Sunshine Books (an imprint of Scot Books Ltd) 12/13 Little Newport Street LONDON WC2H 7PP

Editor: Francisco Lyon de Castro

PUBLICAÇÕES EUROPA-AMÉRICA, LDA.
Apartado 8
2726 MEM MARTINS CODEX
PORTUGAL

Edição n.º 33 097/3995

Execução técnica:
Gráfica Europam, Lda.,
Mem Martins

DAVID LAWRENCE

SPECTRUM FUNCIONAL

uma biblioteca de sub-rotinas e programas práticos

ARTE DE VIVER

PUBLICAÇÕES EUROPA-AMÉRICA

NOTAS PROGRAMÁTICAS

Os programas deste livro foram elaborados com um *interface* especial, construído por L. E. Listings, 1 Leswin Road, London N16, cuja ajuda foi inestimável.

Há dois pontos a salientar nas listagens:

1) Os caracteres gráficos e de inversão foram sublinhados apenas para os manter legíveis.

2) O símbolo ↑ será sempre representado por ∧.

ÍNDICE DOS ASSUNTOS

O conjunto dos programas encontra-se dividido em seis capítulos:

PREFÁCIO

Escrevi este livro para preencher aquilo que me parecia ser uma enorme lacuna. Esta lacuna consiste na inexistência de qualquer trabalho destinado a satisfazer o sonho recente do possuidor de um microcomputador, que é o de que a sua máquina não servirá como um simples brinquedo, nem mesmo uma introdução educativa à idade do silício. Pelo contrário, trata-se de um poderoso instrumento, que assume quaisquer tarefas, abrindo uma variedade de novas possibilidades. A maioria dos livros, ou consiste principalmente em aspectos triviais, ou pretende alcançar um objectivo demasiado grande — talvez mesmo uma capacidade — de experimentação.

Eu queria escrever um livro baseado numa compilação de programas sólidos e sofisticados, abordando áreas como as do armazenamento de dados, finança, cálculo, gráficos, administração doméstica e educação. A discussão de técnicas de programação e de assuntos de âmbito geral ressaltariam dos programas, penso eu, de forma muito aproximada à de um artigo de revista associado a um programa. Espero que encontrem utilidade no aparecimento deste livro, não apenas como uma forma de aprender novas técnicas de programação, mas também como um conjunto de úteis programas, que perspectivam uma vasta gama de aplicações, as quais, de outra forma, apenas ficariam acessíveis aos que tivessem possibilidades de comprar cassetes ou de escrever um programa substancial para si próprios.

O conjunto aqui apresentado divide-se, grosseiramente, em cinco grupos. Tive bastante consciência do perigo de uma simples apresentação massificada de programas indigestos. Por essa razão, bem como pelo facto de que servirão ao *Spectrum,* todos os pro-

gramas foram escritos em forma modular. No texto são dadas informações completas de ensaio de cada módulo de passar ao seguinte.

Uma secção típica do livro apresenta a seguinte forma:

Título do programa: por exemplo, Arquivo de Ficheiros «Spectrum»

Seguem-se informações gerais sobre a respectiva secção, mencionando as vantagens e desvantagens da utilização do *Spectrum* neste tipo de aplicação. Passa-se depois a uma ou mais secções sobre as técnicas específicas de programação usadas nesta aplicação, as quais podem incluir técnicas de busca, disposições para indicadores e dados para armazenagem em sequências dimensionadas. Cada módulo utilizado no programa é, depois, examinado em pormenor.

O texto que acompanha os módulos dispõe-se em três secções: *a*) breve referência aos objectivos do módulo e a quaisquer pontos de interesse levantados; *b*) um comentário às próprias linhas, identificando exemplos de técnicas particulares e apontando as unidades em que, naturalmente, o módulo se inclui; *c*) são dadas sugestões para alguns ensaios simples que poderá gostar de executar, para se assegurar de que, no mínimo, o módulo não se encontra desastrosamente falhado, antes de introduzir o módulo seguinte.

Tome a sério estes ensaios — eles podem evitar-lhe bastantes apertos, mais tarde.

Uma vez fornecida a lista completa dos módulos, passa-se a uma breve revista das principais lições eventualmente aprendidas no decurso da introdução do programa.

Uma das vantagens deste método é a de que os módulos podem sem usados noutro contexto, uma vez compreendida a sua função. Esta possibilidade é ainda mais atractiva pelo facto de o *Spectrum* poder fundir *(merge)* programas.

A capacidade do *Spectrum* para aceitar vários enunciados numa única linha *Basic* pode ser uma bênção ou uma maldição. Por facilidade de utilização e de edição, evitei usar de multienunciado, ao serem elaboradas novas aplicações.

Há, portanto, três formas de utilizar este livro:

1) Como forma inovadora de aprender a fazer programas.
2) Como um conjunto de programas de útil aplicação.

3) Como uma vasta colecção de módulos, ou sub-rotinas, que podem ser utilizados num número ilimitado de outros programas. No final do livro encontrarão um índice, abrangendo as funções de cada um dos módulos de programação.

Por outras palavras, pode tratar este livro como introdução a uma programação mais sofisticada, como biblioteca de programas úteis ou como um manual de referência programática. No entanto, para tirar o máximo rendimento do livro, recomendo que comece pelo princípio e vá avançando e compreendendo.

No que respeita à técnica de aprendizagem, ser-lhe-á útil ler o sumário do livro duas vezes, para ficar com uma ideia do livro no seu todo. Ao abordar cada secção, leia a respectiva introdução, passe por cima do conteúdo e leia o sumário. Leia depois toda a secção. Desta forma, lembrar-se-á de muito mais do conteúdo.

CAPÍTULO 1

ENCONTRAR UMA FICHA — O «SPECTRUM» COMO FICHEIRO

1.1 UNIFICHA

Mais cedo ou mais tarde, a maioria dos possuidores de micros perceberá que o seu novo amigo digital se revela realmente quando começa a armazenar informações, a processá-las e apresentá-las de forma que seria laboriosa, se praticada manualmente. Em breve iniciam a tarefa de escrever programas simples, que guardarão os nomes e moradas dos amigos, e catalogarão o seu álbum de selos. Podem acabar por ter uma meia dúzia de programas, todos eles utilizando, sensivelmente, o mesmo método, embora trabalhando em diferentes tipos de informações. Neste capítulo examinamos a forma como um simples programa pode ser escrito de modo a cobrir uma variedade de tarefas de ficheiro, sem a necessidade de uma constante repetição da escrita, de cada vez que se depara uma nova utilização.

Antes de escrever o programa devemos decidir-nos por uma forma económica de armazenar os dados que queremos arquivar. Os pequenos programas apenas se podem dar ao luxo de alinhavar esta função, já que é provável que façam uso de toda a memória disponível. Um tal programa poderia, por exemplo, apresentar uma disposição *(array)* com dimensões de 50, 4, 20. Isto permitiria uma armazenagem até 50 acessos, cada um constituído por quatro assuntos e cada assunto com um comprimento máximo de 20 caracteres. A vantagem disto é que cada acesso à ficha seria claramente identificável, visto que o acesso X ficaria formado, se a disposição se chamasse **A$**, por A$(X,1), A$(X,2), A$(X,3) e A$(X,4).

A desvantagem deste método é que, para a maioria das utilizações de arquivo, pode resultar um enorme desperdício de espaço dentro da limitada memória disponivel. A raão disto é que, muito simplesmente, o comprimento do espaço fixo atribuído a cada assunto deve ser adequado ao mais longo assunto que poderá vir a ser introduzido. Se, por acaso, pretender arquivar os nomes dos seus amigos e um deles tenha sido baptizado com o apelido Farquarson-Smythe, então, terá de reservar, no minimo, dezassete caracteres para cada apelido, apesar de nenhum dos outros seus amigos ter um apelido sequer comparável a tão impressionante nome. O espaço atribuido à maioria dos nomes ficaria vazio, em mais de metade.

Este é um problema com que depara qualquer método de arquivo que atribua uma quantidade de memória fixa a cada um dos assuntos, sem ter em conta o comprimento. Mas, se não for atribuida uma quantidade fixa de espaço a cada acesso, então, a ficha em que os dados são armazenados não ficará dividida segundo um modelo regular. Isto torna difícil ao programa localizar a posição dos acessos individuais ou mesmo identificar onde um acesso termina e outro começa.

Tomemos como exemplo os seguintes dois acessos de uma ficha a que foi atribuida precisamente a mesma quantidade de espaço:

SMITHJOHN3312556456777HIGH STTHOMASBILL4510956785 1EDEN AV.

Não teve, provavelmente, dificuldade em encontrar os nomes dos dois homens, mas o seu *Spectrum* não está tão familiarizado com apelidos comuns. Mesmo você não se terá apercebido de que cada nome era acompanhado pela idade, número de ficha, número de telefone e número da residência do indivíduo em questão. Como irá o programa identificar cada um destes, dado que nenhum pode variar em extensão?

A resposta a este tipo de problemas é dada, normalmente, através de uma combinação de marcadores e indicadores. Os marcadores são sinais colocados no interior do corpo principal de dados e que permitem ao programa identificar a extensão dos assuntos que compõem um acesso. Os indicadores são colocados, regra geral, fora do corpo principal de dados, consistindo numa lista das posições de todos os acessos dentro da memória, possibilitando ao programa saltar para o meio de uma longa e complexa ficha e encontrar o primeiro carácter do acesso desejado, infalivelmente.

O programa que se segue utiliza marcadores e indicadores para tratar uma ficha constituida por 28000 caracteres, possivelmente constituida por centenas de acessos separados, todos reunidos de forma aparentemente casual. O programa chama-se «Unificha». Espero que venha a desempenhar um valioso papel na sua biblioteca de programas. De maior importância, as técnicas utilizadas são ferramentas essenciais para aqueles que querem atulhar os seus *Spectrum*, de 16K ou 48K, com a máxima quantidade de informações. O programa está dimensionado para ser utilizado numa versão de 48K.

MÓDULO 1.1.1

```
1000 PAPER 7: CLS : BORDER 7: IN
K 6: PAPER 0: PRINT PAPER 2;"
     UNIFILE                  "
1010 PRINT '"FUNCTIONS AVAILABLE
:"
1020 PRINT '"      1)SET UP NEW F
ILE"
1030 PRINT '"      2)ENTER INFORM
ATION"
1040 PRINT '"      3)SEARCH/DISPL
AY/CHANGE"
1050 PRINT '"      4)STOP"
1060 PRINT '''"PLEASE ENTER WHICH
 YOU REQUIRE."
1070 INPUT Z$
1080 CLS
1090 IF Z$="1" THEN GO SUB 1210
1100 IF Z$="2" THEN GO SUB 1440
1110 IF Z$="3" THEN GO SUB 2180
1120 IF Z$="4" THEN GO SUB 1150
1130 CLS
1140 GO TO 1000
1150 PRINT AT 10,5; INK 7; PAPER
 2;"FILING SYSTEM CLOSED"
```

```
1160 BEEP 2,2
1180 INPUT "Have you input new i
nformation  you wish to save? (Y
/N)";0$: IF 0$="N" THEN STOP
1190 SAVE "UNIFILE": PRINT '"Rew
ind tape, then Press any key to
VERIFY": PAUSE 0: VERIFY "UNIFIL
E": STOP
```

Em geral, um programa utilitário que não comece com uma ementa razoavelmente definida das funções disponíveis é um mau programa. Se não concorda com isto, acabará por concordar, eventualmente, quando tiver de retomar um programa complexo, embora útil, com o qual já não trabalhe há muito tempo, vendo-se obrigado a gastar horas a percorrer a listagem até conseguir descobrir o que é necessário fazer para pôr o programa a trabalhar para si.

Neste módulo pede-se ao utilizador que escolha entre quatro funções numeradas. Não são aqui tomadas medidas especiais contra erros de introdução de dados, que não são importantes nesta fase. Se for introduzido um carácter ou número diferente dos de 1 a 4, o programa ignorá-lo-á. O módulo contém igualmente a função STOP, tal como uma das quatro escolhas da ementa. Isto serve o objectivo de determinar um fim à utilização do programa e lembra ao utilizador que deve regravar qualquer novo dado que tenha sido introduzido.

Comentário

Linha 1000: Qualquer programa do *Spectrum* que não se destine a ser usado a preto-e-branco precisa de declarar, algures perto do início, as cores a serem utilizadas para:

a) A margem em volta do *écran*.
b) O *écran*.
c) A tinta em que os caracteres aparecem no *écran*.

Há três diferentes instruções PAPER (papel) nesta linha. A primeira instrução está isolada e, juntamente com o comando CLS

(comando de cor), passa o *écran* a branco. Na segunda instrução, a cor do papel passa a preto, de forma a que a ementa se destaque vivamente do fundo branco. A terceira instrução PAPER está ligada a uma instrução PRINT (impressão), não tendo qualquer influência permanente na cor do papel, embora assegurando que a palavra Unifícha seja impressa sobre fundo vermelho.

É importante que se faça distinção entre os comandos de cor que estão isolados e estabelecem uma cor, e aqueles que se referem apenas à simples informação PRINT à qual estão ligados. Uma única informação PRINT pode incluir todas as características determinadas por esses comandos que se lhe encontram ligados. Por exemplo:

```
PRINT FLASH 1; OVER 1; INVERSE 1; PAPER 7; INK 0; «HELLO»
```

Nunhum destes comandos de cor terá qualquer efeito sobre informações PRINT noutra parte do programa.

Linha 1070: Quando introduzir dados para a ementa de um programa, lembre-se de verificar se as variáveis que utiliza não estão duplicadas em outras partes do programa.

Linha 1080: A ementa é impressa sobre papel branco, em faixas de preto nítido. Mas o último comando PAPER põe o fundo a preto, de modo que este comando CLS põe todo o *écran* a preto, assim permanecendo até que o programa regresse à ementa.

Linha 1190: Para qualquer programa de armazenagem de dados é muito mais conveniente ter uma instrução SAVE (salva), introduzida no programa, do que ter de introduzi-la em modo directo de cada vez que novos dados tenham sido acrescentados.

Ensaio do módulo 1.1.1

Um ensaio rápido e pouco rigoroso resume-se a passá-lo (RUN), introduzindo números de 1 a 3. O programa deverá então parar, reproduzindo 0 OK, seguido do correcto número de linha de 1090 a 1110. Uma entrada de 4 resultará num expedito SAVE e VERIFY (verificação) do programa. Qualquer outra entrada de dados será ignorada.

```
2750 REM **********************
2760 REM FUNCTIONAL SUBROUTINES
2770 REM **********************
2780 INPUT Q$
2790 LET Q$=CHR$ (LEN Q$+1)+Q$
2800 RETURN
2810 PRINT A$(I,2 TO CODE A$(I,1
));":";
2820 RETURN
2830 PRINT FN A$( )(2 TO )
2840 RETURN
2850 FOR I=1 TO X
2860 GO SUB 2810
2870 GO SUB 2830
2880 LET C=C+CODE B$(C)
2890 NEXT I
2900 RETURN
```

Este módulo simples contém algumas rotinas muito breves, que podem ser colocadas de forma mais económica numa sub-rotina, em vez de serem escritas por completo de cada vez que são precisas. De notar aqui a semelhança com a utilização de uma função definida pelo utilizador, que serve uma função similar de economia de espaço. Se uma função deve trabalhar sempre na base das mesmas variáveis, também uma sub-rotina de linha única pode ser igualmente eficiente. As funções definidas entram no que lhes compete quando a mesma função é feita para trabalhar sobre diferentes variáveis, em diferentes sítios.

A linha 2790 podia, por exemplo, ser substituída por uma função definida tal como DEF FN Q$() = CHR$(LEN Q$ + 1) + Q$. A apresentação desta função ocuparia sempre, no entanto, duas linhas, INPUT Q$ e LET Q$ = FN Q$(), não havendo uma real poupança, se comparada com a única linha necessária para evocar a breve sub-rotina de 2780. Se houvesse três ou quatro sequências diferentes em que pretendêssemos aplicar esta função, poderíamos tê-la definido como DEF FN Q$(Q$) = CHR$(LEN Q$ + 1) + Q$.

A função pode agora ser aplicada a outras sequências, colocando apenas a sequência necessária entre parênteses quando a função for evocada. Exemplo: LET C$ = FN Q$(C$). Se quisèssemos trabalhar em C$ com uma sub-rotina de linha única, precisaríamos de uma sub-rotina extra para tratar de C$.

A moral de tudo isto resume-se a: definir funções apenas por amor a elas pode ser uma perda de tempo. Guarde funções definidas valiosas para operações que possam ser aplicadas a diferentes variáveis, em diferentes sítios.

O módulo é formado de quatro sub-rotinas, tal como segue:

1) Linhas 2780-2800. Esta secção acrescenta à entrada Q$ o marcador mencionado na introdução. O marcador assume a forma de um único carácter. Lembre-se de que, no *Spectrum*, cada carácter tem um valor de código único; pode encontrar-se uma lista destes valores do Apêndice A do manual do *Spectrum*. A função CHR$ pode ser usada para seleccionar o carácter correcto para cada valor entre 0 e 255, enquanto a função CODE (código) transfere qualquer carácter para um valor entre 0 e 255.

Utilizando estas duas funções, é possível armazenar valores entre 0 e 255 num único carácter. No caso dos nossos marcadores, o carácter único que é acrescentado armazena a extensão da sequência, mais uma para o próprio marcador, de forma que, quando a sequência é incluída no principal ficheiro de dados, o marcador pode ser utilizado para identificar quanto daquilo que vem depois do marcador faz parte do mesmo assunto. Se o marcador tem um valor de 11, então o assunto consiste no marcador e nos 10 caracteres seguintes.

2) Linhas 2810-2820. Estas linhas imprimem nomes de assuntos, tais como nome e morada. De notar que o valor do marcador é aqui utilizado para extrair a parte útil de uma linha, dentro de uma disposição. Os nomes dos assuntos são armazenados em A$, cujas linhas têm vinte caracteres de extensão. A diferença entre a extensão do nome do assunto e a extensão da linha dentro da disposição é constituída por espaços que não desejamos imprimir.

A linha 2810 imprime apenas aquela parte da linha relevante em A$ que contém os caracteres do nome do assunto. Nem o marcador nem os espaços são impressos. Isto pode ser uma poderosa

ajuda na formatação, quando se armazena texto em disposições que são maiores do que o texto.

3) Linhas 2830-2840. FN A$ extrai um único assunto do principal ficheiro de dados, sendo explicado no próximo módulo.

4) 2850-2900. Esta sub-rotina é usada para imprimir acessos da ficha. As variáveis utilizadas serão explicadas na discussão de módulos posteriores.

Ensaio do módulo 1.1.2

O correcto desempenho destas sub-rotinas apenas pode ser efectivamente ensaiado quando tiverem sido introduzidos novos módulos.

MÓDULO 1.1.3

```
1200 REM ************************
1210 REM ENTRY STRUCTURE
1220 REM ************************
1230 PRINT PAPER 2;"          FILE
 STRUCTURE                "
1240 PRINT '"HOW MANY ITEMS IN E
ACH ENTRY?"
1250 INPUT X
1260 CLS
1270 DIM A$(X,20)
1280 PRINT PAPER 2;"          NAMES
 OF ITEMS                 "
1290 FOR I=1 TO X
1300 PRINT "ITEM ";I;":";
1310 GO SUB 2780
1320 PRINT Q$(2 TO )
1330 LET A$(I)=Q$
1340 NEXT I
```



```
1350 DIM B$(28000)
1360 LET B$(1 TO 4)=CHR$ 2+CHR$
0+CHR$ 2+CHR$ 255
1370 DEF FN A()=256*CODE Y$(2*S-
1)+CODE Y$(2*S)
1380 DEF FN A$()=B$(C TO C+CODE
B$(C)-1)
1390 LET P=5
1400 LET Y$=CHR$ 0+CHR$ 1+CHR$ 0
+CHR$ 3
1410 LET N=2
1420 RETURN
```

É este o módulo que permite ao «Unificha» assumir diferentes formas, de acordo com o capricho do utilizador. No decurso do módulo, as principais disposições e variáveis são colocadas a postos para os dados que hão-de vir. De salientar que uma das consequências disto é que nenhum dos dados previamente arquivados se perderá. Não discutiremos aqui, em pormenor, a utilização das várias disposições, preferindo adiar essa tarefa até que comecemos realmente a utilizá-las.

Comentário

Linhas 1230-1340. Um acesso típico à ficha poderia consistir em nome, morada, idade e número de telefone. No decurso destas linhas, o programa regista a quantidade destes assuntos em cada acesso da variável X. Os nomes dos assuntos são solicitados e armazenados na disposição A$, tendo sido destacado um marcador pela sub-rotina da linha 2780. Note-se que imprimimos Q$ despojado do seu primeiro carácter, visto que o marcador não é um carácter com significado.

A linha 1350 é a principal disposição em que serão armazenados os acessos.

A linha 1360 organiza dois acessos sem significado, que ficarão a marcar o princípio e fim da ficha.

Linhas 1370-1380. Dois exemplos de funções definidas pelo utilizador, que poderiam igualmente ser substituídas por sub-rotinas de uma só linha. A primeira função extrai o valor de um indicador, sendo explicada no decurso do módulo 5. A segunda função extrai um único assunto da ficha principal, baseada no valor do marcador que se encontra na posição C da ficha.

Linha 1390. P é a variável usada para registar o primeiro espaço vazio em B$. B$ terá sempre 28 000 caracteres de extensão, embora nós utilizemos somente uma parte. É claro que precisamos de saber a quantidade já utilizada.

Linha 1400. Y$ armazena os indicadores sob a forma de códigos de caracteres, método que é abordado em relação ao módulo 5.

Linha 1410. N é a variável que regista o número de acessos da ficha.

Ensaio do módulo 1.1.3

Podemos agora ensaiar os módulos 2 e 3. Passe o programa e seleccione a função 1 da ementa. Deverá poder especificar alguns assuntos, atribuindo-lhes nomes. Depois de ter feito isto, pare o programa e, em modo directo, imprima as várias disposições e variáveis, do seguinte modo:

B$: ??? COPY
Y$: ????
N: 2
P: 5

X deverá igualar o número de assuntos especificados e a disposição A$ deverá incluir as linhas X, cada uma delas contendo um nome de um assunto e um marcador pregado logo à cabeça.


```
1430 REM ************************
1440 REM NORMAL INPUT
1450 REM ************************
1460 LET R$=""
1470 PRINT PAPER 2;"          EN
TRIES                "
1480 PRINT '"COMMANDS AVAILABLE:
"
1490 PRINT '"(ENTER ITEM SPECIFI
ED)"'"(""ZZZ"") TO QUIT"
1500 PRINT "********************
************"
1510 PRINT "FILE SIZE:";P-1;"/";
LEN B$
1520 FOR I=1 TO X
1530 GO SUB 2810
1540 GO SUB 2780
1580 PRINT Q$(2 TO )
1590 IF Q$(2 TO )="ZZZ" THEN RET
URN
1600 LET R$=R$+Q$
1610 NEXT I
1620 CLS
1630 GO SUB 1660
1640 GO TO 1440
```

O objectivo deste módulo é aceitar a entrada de um acesso composto pelo número correcto de assuntos, apresentando esse acesso na secção do programa que o introduzirá no sítio correcto da ficha.

Comentário

Linha 1600. R$ é o acesso e compõe-se de vários Q$ acrescentados em conjunto.

Ensaio do módulo 1.1.4

Se já introduziu alguns nomes sensatos de assuntos, comece então o programa com GOTO 1 e solicite a função 2 da ementa. O programa deverá pedir-lhe uma introdução de dados para cada nome de assunto. Depois do número correcto de nomes de assuntos, o programa pára, dando a informação 0 OK, 1630:1. A dimensão da ficha deverá ser de 4/28 000 e, se imprimir R$, verá aparecer os seus assuntos, cada um deles precedido de um carácter marcador.

MÓDULO 1.1.5

```
1650 REM ************************
1660 REM PLACE DATA IN FILE
1670 REM ************************
1680 IF P+LEN R$-1<LEN B$ THEN G
O TO 1730
1690 PRINT AT 14,10;"FILE NOW FU
LL"
1700 PRINT '''   Press any key t
o continue"
1710 PAUSE 0
1720 RETURN
1730 LET POWER=INT (LN (N-1)/LN
2)
1740 LET S=2^POWER
1750 LET T$=R$(2 TO CODE R$(1))
1760 FOR K=POWER-1 TO 0 STEP -1
1770 LET C=FN A()
1780 LET U$=FN A$()(2 TO )
1790 LET S=S+(2^K)*(T$>U$)-(2^K)
*(T$<U$)
1810 IF S>N-1 THEN LET S=N-1
1820 IF S<2 THEN LET S=2
1830 NEXT K
1840 LET C=FN A()
1850 LET U$=FN A$()(2 TO )
```



```
1860 IF T$<U$ THEN LET S=S-1
1870 LET B$(P TO P+LEN R$-1)=R$
1880 LET N=N+1
1890 LET Y$=Y$(1 TO 2*S)+CHR$ IN
T (P/256)+CHR$ (P-256*INT (P/256
))+Y$(2*(S+1)-1 TO )
1900 LET P=P+LEN R$
1910 RETURN
```

Este módulo é o mais complexo do programa. Antes de passarmos a um comentário pormenorizado, discutiremos dois pontos:

1) Utilização das sequências *(strings)* para armazenar números.
2) Técnica da busca binária.

Números em sequências

Vimos já, no módulo 3, que os indicadores para o nosso programa estão armazenados numa sequência, Y$. Poderá perguntar-se por que razão os valores numéricos não são armazenados numa disposição numérica a direito. A resposta apoia-se na economia de memória e de tempo, sendo esta a mais significativa. Vejamos primeiro a economia de memória.

Para fazer face ao máximo número de acessos que é provável encontrar, teria de se declarar uma disposição numérica para os indicadores composta por algo como 2000 elementos. É muito improvável que você venha a ter 2000 acessos, embora possivel. Teria sérios problemas se a sua disposição se revelasse com falta de espaço para o número de acessos que têm de ser indicados. Uma disposição não pode ser redimensionada sem que se perca tudo o que ela encerra. O verdadeiro problema é que uma disposição numérica de 2000 elementos ocuparia cerca de 10 000 *bytes* de memória, devido à forma como o *Basic* Sinclair armazena os números. Isto é uma proporção extravagante do total de memória disponivel.

O *Spectrum* confere cinco *bytes* de memória para cada número armazenado numa disposição, num esforço para cobrir a maior gama de números possivel... de facto, até 4 294 967 295. Não iremos precisar de nada que se compare com isto: a nossa ficha tem apenas 28 000 caracteres de extensão, pelo que apenas precisaremos de nú-

meros inteiros entre 1 e 28 000. É possível representar estes números utilizando sequências de dois caracteres.

Cada carácter tem um valor de código que lhe é único, entre 0 e 255. Pode usar-se um só carácter para armazenar qualquer valor entre 0 e 255, bastando utilizar o carácter que tenha esse código. Assim, o carácter A representa o número 65, e a palavra-chave GO-TO — apenas mais um carácter, no que diz respeito ao *Spectrum* — representa o número 236. Números maiores que 255 são representados utilizando o segundo carácter, que armazena o número de 256 inteiros, de forma muito aproximada à que, no nosso sistema decimal, em 36, o 3 representa três dezenas inteiras. Dois caracteres dão-nos, portanto, a possibilidade de armazenar qualquer número inteiro positivo até 255*256+255, o que equivale a 65 535 e é mais do que a suficiente para fazer face à nossa ficha de 28 000 caracteres.

Desde que apenas deseje números positivos inteiros entre 0 e 65 535, é possível poupar três dos cinco *bytes* que o *Spectrum* usaria se os mesmos números fossem armazenados numa disposição numérica.

A vantagem é que dois dos três *bytes* poupados são deitados fora, na busca da rapidez.

Imagine novamente a nossa disposição numérica de 2000 elementos e imagine que pretende acrescentar ou apagar um número, algures perto do princípio. Se se apagar simplesmente um valor indesejado, ficará um buraco, ou antes, um zero, no local onde antes estava o número. Se inserir um número, irá escrever por cima do que já lá estiver. Para evitar qualquer destes dois resultados indesejados, terá de se certificar de que todos os elementos da disposição podem ser deslocados para cima ou para baixo num lugar. Se a posição em que pretende inserir um número novo for a posição 1, então 1999 números terão de ser afastados para dar lugar ao primeiro. Isto pode ser feito com três linhas de *Basic* sob a forma de um simples desvio, embora leve tempo, especialmente num *Spectrum*. Nem mesmo os melhores amigos deste descrevem o *Spectrum* como deslumbrantemente rápido.

Compare agora esse desvio, repetindo a sua operação 1999 vezes, com o que se segue:

```
LET A$=«XX» + A$
```


Utilizando o soberbo tratamento das sequências pelo *Spectrum*, podemos simplesmente inserir dois *bytes* no início, no fim ou no meio de um sequência com uma só instrução. Isto é bastante rápido, mas tem uma contrapartida... duplica momentaneamente a quantidade de espaço ocupado pela sequência. O *Spectrum* precisa de reter na memória, ainda que por um momento, a nova A$ que a linha cria, juntamente com a velha A$ que está a ser utilizada para criar aquela. Esta limitação é uma das maiores contrapartidas em relação à capacidade de tratamento de sequências pelo *Spectrum*, sendo difícil de evitar. Isto significa que Y$, aqui utilizada para armazenar os pares de caracteres utilizados como indicadores dos acessos na ficha principal, é realmente duas vezes maior do que parece, já que, na nossa busca de rapidez, duplicará momentaneamente de cada vez que lhe quisermos acrescentar ou retirar algo. A duplicação pode ser apenas momentânea, mas, ainda assim, temos de lhe conceder espaço na memória. É uma pena, mas temos de preencher a viver com este facto.

Esta contrapartida é a razão por que não utilizamos o mesmo método para inserir ou apagar dados na nossa ficha principal B$. A fazê-lo, reduziríamos à metade o espaço disponível para acessos. Organizámos B$ como uma disposição fixa e, sempre que quisermos apagar alguma coisa, movimentamos para baixo o resto da ficha, pedaço a pedaço, para preencher o intervalo criado.

Busca binária

Utilizaremos a técnica de busca binária para reduzir o número possível de comparações feitas ao encontrar o sítio correcto para inserir um novo acesso, de entre os 12 500 a 15 000 possíveis. Considere-se o exemplo seguinte.

Estabelecemos uma ficha que contém agora 2000 assuntos e a presente entrada precisa de ser inserida na posição 1731, embora o programa ainda não tenha descoberto isto. O programa começa a sua busca procurando no primeiro acesso e comparando-o com o novo acesso a ser inserido. Conclui que o novo acesso é o maior dos dois, pelo que passa a compará-lo com o acesso número 2. Eventualmente, após 1731 comparações, o programa encontra o primeiro acesso da ficha que é maior do que o novo acesso. Está descoberta a posição correcta para o assunto.

Compare este processo sem rodeios com aquele que se segue, para uma ficha das mesmas dimensões e uma inserção na mesma posição.

O programa começa por examinar o acesso na posição 1024, sendo essa a potência de 2 que é menor ou igual ao número de acessos na ficha. Verifica-se que o acesso nessa posição é menor do que o novo acesso. O programa adiciona 1024/2 a 1024, resultando 1536. O acesso em 1536 continua a ser menor do que o novo acesso. Assim, 1024/4 é adicionado a 1536, dando 1792. O acesso em 1792 é maior do que o novo acesso, por isso, 1024/8 é subtraido de 1792, dando 1664. A busca prossegue nos locais seguintes da ficha, com as adições e subtracções que se seguem.

1644 (depois adiciona 64)
1728 (depois acrescenta 32)
1760 (depois subtrai 16)
1744 (depois subtrai 8)
1736 (depois subtrai 4)
1732 (depois subtrai 2)
1730 (depois adiciona 1)
Resultado final: 1731

A eficiência de uma busca binária é evidente.

Comentário

Linhas 1680-1720. Estas linhas providenciam espaço na ficha para o novo acesso.

Linhas 1730-1830. A busca binária aplica-se aos acessos em B$. A busca é conduzida com base na ordem alfabética do primeiro assunto em cada acesso. Para uma explicação de como o *Spectrum* entende a ordem alfabética, ver a p. 95 do manual do *Spectrum*.

A linha 1730 encontra a potência máxima de 2, que continua a ser menor ou igual ao número de acessos da ficha, fazendo uso da função logaritmica. A posição de busca é colocada em valor igual ao daquela potência.

Linha 1750. T$ é criada em igualdade com o primeiro assunto do acesso à posição de busca.

Linhas 1760-1830. Este desvio acrescenta ou subtrai potências de 2, segundo os princípios estabelecidos na discussão da escolha binária.

Linha 1770. FN A foi definida na linha 1370. De dois caracteres em Y$, extrai um valor numérico que é um indicador do primeiro carácter de um acesso na ficha principal.

Linha 1780. FN A$ foi definida na linha 1380. Extrai das fichas principais o assunto cujo marcador se encontra na posição C, em B$.

Linha 1790. Esta linha precisa de mais explicações. Uma condição tal como T$ > U$ pode ser verdadeira ou falsa, mas, na utilização do dia-a-dia, não pode dizer-se que tenha um valor em termos idênticos ao de um número ou de uma variável. Para o *Spectrum*, no entanto, T$ > U$ tem um valor real, que, ou é 1, caso a condição seja verdadeira, ou 0, se a condição for falsa. O valor da condição pode ser usado num programa do mesmo modo que um número ou uma variável. Nesta linha, em particular, se T$ > U$, a condição terá o valor de 1 e a S será adicionado (2↑K)*1. Por outro lado, T$ > U$ será igual a 0, sendo S subtraido de (2↑K)*0. Se T$ tivesse sido menor que U$, então os papéis inverter-se-iam. Se T$ tivesse sido igual a T$, ambas as condições teriam sido falsas e S não se teria alterado.

Linhas 1810-1820. Se S, a posição de busca, aponta para um dos acessos fictícios, estas duas linhas enviam-no para trás, para o corpo principal de dados.

Linhas 1840-1850. Tendo completado a busca binária, o assunto na posição seleccionada é extraído para ser examinado. Se o assunto nesta posição e o novo assunto forem iguais, este último será numerado após o assunto existente. Caso não sejam iguais, o novo assunto será numerado antes do já existente.

Linha 1870. O novo acesso é acrescentado ao fim da ficha. A ordem correcta dos acessos na ficha é guardada apenas em Y$. Desde que Y$ saiba onde se encontra, por exemplo, o 378.° assunto, não tem importância que esteja realmente armazenado no 378.° lugar.

Ensaio do módulo 1.1.5

É difícil ensaiar este módulo até ter sido acrescentada ao programa a busca e função de apresentação, permitindo que se apresentem com facilidade os acessos. Pode dar-se ao cuidado de introduzir alguns acessos, parando depois o programa para procurar saber se eles foram inseridos em B$. Lembre-se de que eles foram inseridos pela ordem em que foram introduzidos no programa. Com este desvio, pode igualmente examinar Y$.

```
9000 FOR S=1 TO LEN Y$ STEP 2: PRINT FN A(): NEXT S
```

Isto imprimirá os valores dos indicadores, que poderá comparar com os índices dos acessos na ficha principal.

MÓDULO 1.1.6

```
2170 REM ************************
2180 REM SEARCH
2190 REM ************************
2200 LET S=2
2210 PRINT PAPER 2;"
SEARCH                "
2220 PRINT '''"COMMANDS AVAILABLE
:"
2230 PRINT ">INPUT ITEM FOR NORM
AL SEARCH"'">PRECEDE WITH ""SSS"
" FOR"'" SPECIAL SEARCH"'">PRECE
DE WITH ""III"" TO SEARCH"'" FOR
 FIRST CHARACTER OF ENTRY"'">""E
NTER"" FOR FIRST ITEM ON FILE"
2240 PRINT "**********************
************"
2250 PRINT '''"INPUT SEARCH ITEM:
";
2260 GO SUB 2780
2270 PRINT Q$(2 TO )
2280 LET S$=Q$
```



```
2290 IF LEN S$=1 THEN GO TO 2510
2300 LET C=FN A()
2310 IF LEN S$<5 THEN GO TO 2430
2320 IF S$(2 TO 4)<>"III" THEN G
O TO 2390
2330 FOR I=S TO N
2340 LET S=I
2350 LET C=FN A()
2360 IF B$(C+1)=S$(5) THEN GO TO
 2510
2370 NEXT I
2380 RETURN
2390 IF S$(2 TO 4)<>"SSS" THEN G
O TO 2430
2400 GO SUB 2920
2410 IF C4=1 THEN GO TO 2510
2420 RETURN
2430 FOR I=1 TO X
2440 IF FN A$()=S$ THEN GO TO 25
10
2450 IF FN A$()=CHR$ 2+CHR$ 255
THEN RETURN
2460 LET C=C+CODE B$(C)
2470 NEXT I
2480 LET S=S+1
2490 LET C=FN A()
2500 GO TO 2430
2510 LET C=FN A()
2520 LET C4=0
2530 IF FN A$()=CHR$ 2+CHR$ 255
THEN RETURN
2540 CLS
2550 PRINT "ENTRY ";S-1;":-"
2560 GO SUB 2850
2570 LET S=S+1
2580 PRINT AT 16,0; PAPER 2;"
       SEARCH                "
2590 PRINT "COMMANDS AVAILABLE:"
2600 PRINT ">""ENTER"" TO DISPLA
```

```
Y NEXT ITEM"'">""ZZZ"" TO QUIT F
UNCTION"'">""AAA"" TO AMEND"'">"
"CCC"" TO CONTINUE SEARCH"
2610 INPUT P$
2620 CLS
2630 IF P$="CCC" THEN GO TO 2300
2640 IF P$="" THEN GO TO 2510
2650 IF P$<>"AAA" THEN GO TO 271
0
2660 LET C=FN A( )
2670 CLS
2680 GO SUB 1930
2710 IF P$="ZZZ" THEN RETURN
2720 IF P$="AAA" THEN RETURN
2730 CLS
2740 GO TO 2260
```

O objectivo deste módulo é apresentar acessos da ficha, quer um de cada vez, desde o principio, quer começando no primeiro acesso que satisfaça certas condições de busca. Tendo apresentado um acesso, o módulo dá ao utilizador a possibilidade de continuar a busca, de examinar o acesso seguinte, de mudar o acesso ou de o apagar da ficha. Saliente-se o uso contínuo de FN A e de FN A$ para fornecer o local de um acesso e para o extrair da ficha.

Comentário

Linha 2200. S é o número do acesso que estiver a ser examinado no momento. Inicialmente é colocada em 2, pois o primeiro acesso é fictício.

Linhas 2290-2380. Se o utilizador introduzir uma instrução de busca começada por III, o programa examina o primeiro assunto de cada acesso até encontrar um começado pelo carácter e seguido de III. Se não for encontrado qualquer assunto nestas condições, o programa regressa à ementa principal.

Linhas 2390-2420. A busca especial, que procura qualquer combinação especifica de caracteres, sem reparar se se trata de um assunto completo ou não, é levada a cabo por uma rotina separada, que é solicitada por estas linhas, se a instrução de busca começar por SSS.

Linhas 2430-2500. Os assuntos completos da ficha são examinados, para se ver se correspondem ao assunto que o programa busca. Isto é muito mais rápido do que a busca especial, que se desloca ao longo da ficha carácter por carácter. Não poderá ser usada uma rápida busca binária, dado que apenas os primeiros assuntos de cada entrada estão por ordem alfabética. Para que esta busca tenha sucesso, a introdução do assunto deverá ser igual à do assunto em memória. Se se procurasse Smith,J na ficha, não se encontraria Smith, John, ao passo que, utilizando uma busca especial, SSSSmith,J encontraria Smith,J ou Smith,John, mas seria muito mais demorado.

Linhas 2510-2570. Esta secção imprime um acesso que utiliza a sub-rotina em 2850, que já examinâmos.

Linhas 2580-2740. Tendo descoberto um acesso que satisfaz o critério da busca, o módulo oferece agora ao utilizador a possibilidade de passar a ficha página a página, acesso por acesso, procurando o próximo acesso que satisfaça o critério original de busca, ou solicitar a rotina que permite alterar ou apagar o acesso.

Ensaio do módulo 1.1.6

Pode ensaiar o correcto funcionamento de todas as funções de busca, com excepção da busca especial. A função de correcção ainda não foi introduzida.

MÓDULO 1.1.7

```
2910 REM ************************
2920 REM SPECIAL SEARCH
2930 REM ************************
2940 LET C4=0
2950 FOR H=S TO N-1
2960 LET S=H
2970 LET C=FN A( )
2980 LET C1=C
2990 FOR I=1 TO X
3000 LET C1=C1+CODE B$(C1)
3010 NEXT I
3020 FOR J=C+1 TO C1-LEN S$+5
3030 IF B$(J TO J+LEN S$-5)<>S$(
5 TO ) THEN GO TO 3060
3040 LET C4=1
3050 RETURN
3060 NEXT J
3070 NEXT H
3080 LET C4=0
3090 RETURN
```

Este módulo contém a rotina de busca especial acima mencionada.

Comentário

Linha 2940. C4 é o marcador utilizado para dar a conhecer se, regressando ao módulo 6, a combinação específica de caracteres foi encontrada.

Linhas 2980-3010. Em primeiro lugar, C1 é igualada ao endereço inicial do acesso a ser examinado. Os marcadores ligados aos assuntos X do acesso são acrescentados a C1, tornando C1 igual ao endereço inicial do acesso seguinte. Repare-se que estamos agora a falar do endereço inicial do acesso seguinte na ficha principal e não do acesso seguinte por ordem alfabética.

Linhas 3020-3060. O acesso é examinado, carácter por carácter, para comparação com a combinação de caracteres específicos da instrução de busca.

Ensaio do módulo 1.1.7

Introduza uma série de combinações de caracteres, alguns dos quais estão presentes na ficha e outros não. Não se esqueça de os preceder de SSS.

MÓDULO 1.1.8

```
1920 REM ************************
1930 REM CHANGE ENTRY
1940 REM ************************
1950 LET S=S-1
1960 LET C=FN A( )
1970 LET R$=""
1980 PRINT "ENTRY ";S-1;":-"
1990 FOR I=1 TO X
2000 GO SUB 2810
2010 GO SUB 2830
2020 PRINT AT 17,0; PAPER 2;"
       AMEND                    "
2030 PRINT "COMMANDS AVAILABLE:"
2040 PRINT ">""ENTER"" LEAVES IT
EM UNCHANGED""">""ZZZ"" DELETES
WHOLE ENTRY""">ENTER NEW ITEM"
2050 GO SUB 2780
2060 IF LEN Q$=1 THEN LET R$=R$+
B$(C TO C+CODE B$(C)-1)
2070 LET C=C+CODE B$(C)
2080 CLS
2090 IF LEN Q$=1 THEN GO TO 2120
2100 IF Q$(2 TO )="ZZZ" THEN GO
TO 2130
```

```
2110 LET R$=R$+Q$
2120 NEXT I
2130 GO SUB 3130
2140 IF Q$(2 TO )="ZZZ" THEN RET
URN
2150 GO SUB 1660
2160 RETURN
```

Este módulo dá ao utilizador a possibilidade de mudar ou apagar o acesso apresentado ao módulo pela função de busca.

Comentário

Linhas 2050-2130. Deve lembrar-se de que, no módulo 4, foram construidos novos acessos sob a forma de R$. Nestas linhas, cria-se uma R$ modificada, formada por assuntos tirados directamente do acesso na ficha, ou por assuntos introduzidos para substituir os originais. O acesso original é então apagado da ficha, solicitando a sub-rotina na linha 3130.

Linha 2140. Se o utilizador não especificou que o acesso devia ser apagado, o acesso modificado, sob a forma de R$, é apresentado ao módulo 5, que o insere no local correcto.

Ensaio do módulo 1.1.8

Para se ensaiar completamente este módulo, deve esperar-se o acesso do módulo seguinte, embora possa ensaiar se o módulo apresenta realmente o acesso seleccionado, assunto por assunto, e se quaisquer alterações introduzidas estão registadas em R$. Depois de apresentar todos os assuntos do acesso, o programa pára, dando a indicação 0 OK,2130:1.


```
3100 REM ************************
3110 REM TELESCOPE FILE
3120 REM ************************
3130 LET C=FN A( )
3140 LET SHIFT=1000
3150 LET C1=C
3160 LET C3=C
3170 FOR I=1 TO X
3180 LET C1=C1+CODE B$(C1)
3190 NEXT I
3200 LET C2=C1-C
3210 FOR I=C1 TO LEN B$-1 STEP S
HIFT
3220 IF LEN B$-I+1<SHIFT THEN LE
T SHIFT=LEN B$-I+1
3230 LET S$=B$(I TO I+SHIFT-1)
3240 LET B$(C TO C+SHIFT-1)=S$
3250 LET C=C+SHIFT
3260 NEXT I
3270 LET Y$=Y$(1 TO 2*(S-1))+Y$(
2*(S+1)-1 TO )
3280 FOR I=1 TO N-1
3290 LET S=I
3300 LET C=FN A( )
3310 IF C<=C3 THEN GO TO 3350
3320 LET C=C-C2
3330 LET Y$(2*I-1)=CHR$ INT (C/2
56)
3340 LET Y$(2*I)=CHR$ (C-256*INT
 (C/256))
3350 NEXT I
3360 LET P=P-C2
3370 LET N=N-1
3380 RETURN
```

Sempre que um acesso é apagado da ficha, deixa um espaço que deve ser preenchido. A função deste módulo é apagar um acesso específico, fazendo a ficha alcançá-lo à distância. A ficha não é passada acesso por acesso, mas sim em pedaços de 1000 caracteres.

Comentário

Linha 3170. Este desvio coloca C1 no endereço inicial do acesso seguinte.

Linhas 3210-3260. Este desvio desloca pedaços da ficha com 1000 caracteres ao longo do acesso a ser apagado, começando o primeiro pedaço em C1.

Linhas 3230-3240. Se B$ tivesse sido mencionado em ambos os lados de uma equação —por exemplo: LET B$(C TO C+SHIFT—1)=B$(1 TO 1+SHIFT—1)—, ter-se-ia criado uma sombra momentânea de B$ e o programa teria saído da memória.

Linha 3270. O indicador do acesso apagado é removido.

Linhas 3280-3350. Todos os acessos que foram deslocados ao longo da ficha devem agora ver os seus indicadores corrigidos, visto que os endereços iniciais estão diferentes.

Ensaio do módulo 1.1.9

Utilize a função AMEND (correcção) para apagar um ou dois assuntos, mudando depois os primeiros assuntos e alguns acessos, de forma que tenham de ser deslocados dentro da ficha. Após cada mudança ou eliminação, utilize a função SEARCH (busca) para verificar se a ficha continua na ordem correcta. Se os ensaios forem satisfatórios, o programa está completo.

Sumário

Acabou de completar a introdução de dados de um programa substancial e complexo que, assim o espero, lhe será útil em variadas circunstâncias. Mais importante do que isto, no entanto, é que

tenha aprendido várias técnicas que lhe serão úteis em programas futuros que tenham de tratar, de forma económica, grandes quantidades de dados não numéricos.

Aprendeu aqui a forma de estruturar dados compactos utilizando indicadores e marcadores. Viu como as sequências podem ser usadas com eficiência para armazenar uma gama limitada de dados numéricos. Possui um exemplo funcional da poderosa técnica de busca binária.

Melhor ainda: se se deu ao trabalho de compreender tudo aquilo que introduziu, ficou mais confiante de que vastos e complexos conjuntos de dados podem ser processados sem que tudo degenere em caos... Afinal, grande parte da arte da programação consiste em ter a ousadia de juntar aplicações que parecem desesperadamente complicadas, juntamente com a perseverança de prosseguir a tarefa até ao fim.

Avançando mais

Se compreendeu aquilo que introduziu, talvez lhe agrade abraçar algumas das seguintes tarefas:

1) O programa está deliberadamente escrito com poucas linhas multienunciado. Logo que o programa esteja funcional, seria uma boa ideia tentar encurtá-lo pela combinação de linhas... Aprenderá bastante acerca dos poderes e fraquezas das linhas multienunciado.

2) Já referi que não se faz qualquer uso da busca binária no presente módulo de busca. Por que não acrescentar outra instrução de busca apenas referente ao primeiro assunto de cada acesso, a qual utilizará a rotina de busca binária para efectuar a busca?

3) Tal como se encontra estruturado, o programa não poderá abarcar fichas ou acessos que tenham um número variável de assuntos. Este tipo de estrutura é bastante comum. Como exemplo: uma receita com um título, número variável de ingredientes e instruções. É bastante simples alterar o programa de forma que funcione com três assuntos por cada acesso, embora com o segundo assunto dividido em alguns subassuntos. A função AMEND deverá conseguir apagar ou acrescentar subassuntos.

4) O programa não prevê o envio de acessos a uma impressora *ZX*... o que pode facilmente ser rectificado.

CAPÍTULO 2

ELABORAÇÃO DE UM ORÇAMENTO — O «SPECTRUM» COMO BANQUEIRO

No capítulo anterior examinámos uma variedade de técnicas necessárias ao tratamento de grandes quantidades de dados não numéricos. Neste capítulo trataremos de números e, em particular, de dinheiro.

Neste capítulo são apresentados três programas. O primeiro, «Orçamento», é um instrumento de planificação das finanças da família, um programa que lhe permite examinar as consequências de decisões financeiras complexas e ter uma ideia de como irão estar as finanças nos próximos doze meses. A este programa segue-se o «Contabilista», um programa simples que gera um conjunto de contas para si, em formato legível. Finalmente, «Banqueiro» é um programa que lhe permitirá manter-se ao corrente da sua conta bancária, sem esperar que o banco lhe envie o extracto.

2.1 ORÇAMENTO

Trata-se de um programa substancial, consideravelmente maior do que o «Unificha». Isto poderá ser uma surpresa, já que a quantidade de informação tratada pelo programa é consideravelmente menor que os 28 000 caracteres do «Unificha». Existem três razões principais para este facto. Em primeiro lugar, «Orçamento» é um programa mais flexível em termos daquilo que permite ser fei-

to pelo utilizador. A lista de funções programáticas é bastante mais extensa. Em segundo lugar há uma quantidade bastante razoável de cálculos repetitivos a fazer, o que requer espaço. Em terceiro lugar vem a questão do formato. As informações armazenadas pelo programa seriam indigestas se não fosse o facto de uma grande parte do programa estar dedicado a assegurar que os dados sejam apresentados com clareza.

MÓDULO 2.1.1

```
3330 REM **********************
3340 REM QUESTIONS
3350 REM **********************
3360 PRINT AT P1,0;O$
3370 PRINT AT P1,P2;P$
3375 IF H=2 THEN PRINT AT 21,29;
"(H)"
3380 INPUT Q$
3390 PRINT AT 20,0;O$;O$
3400 PRINT AT 20,0;">>";Q$;"<<"
3410 PRINT AT 21,0;""ENTER"" to
 confirm"
3420 INPUT R$
3430 PRINT AT 20,0;O$;O$
3440 IF R$="" THEN GO TO 3460
3450 GO TO 3360
3460 LET P2=0
3470 DIM T$(9)
3480 LET T$=Q$
3490 RETURN
```

Esta pode parecer uma estranha forma de começar, mas poucos dos outros programas funcionarão até que este tenha sido inserido. O objectivo do módulo é tratar quase todos os avisos necessários no decorrer do programa. Isto possibilita um método flexível de formatação desses avisos e um processo simples de detecção de erros.

Alguns programas avançam até muito longe como protecção contra inserção de dados sem sentido, que desfariam o programa ou tornariam indecifráveis os resultados. Estas técnicas podem contribuir para a robustez do programa, mas não podem protegê-lo de todas as contingências. Só a experiência com o programa a ser utilizado pode determinar se valerá a pena o esforço de programar tais verificações.

Neste módulo adoptamos a simples técnica de relembrar ao utilizador aquilo que foi inserido, pedindo confirmação antes de aceitar os dados. A razão por que temos um verificador de erros neste programa, quando tal não acontecia no «Unificha», deve-se ao facto de ser muito mais fácil cometer um erro ao introduzir uma série de números, não se apercebendo de que tal aconteceu. Isto é muito mais sério do que um erro de ortografia de um nome na inserção de um programa de ficheiro.

Comentário

Linha 3360. P1 é apenas o número de uma linha, o qual é declarado antes de esta rotina ser solicitada. O$ é uma linha de 32 espaços vazios: assegura que a linha onde será impresso o aviso está limpa.

Linha 3370. P2 é a posição ao longo da linha. No início do programa é colocada em zero — o início da linha — e aí permanece a menos que seja alterada antes de esta rotina ser solicitada. P$ é o texto do aviso corrente.

Linhas 3390-3450. A inserção é reimpressa e pede-se ao utilizador que confirme se a inserção é a desejada.

Linha 3460. No caso de P2 deter outro valor que não 0, será reduzido a este número.

Linhas 3470-3480. Em alguns casos, a inserção será comparada com uma sequência armazenada numa disposição, como no caso dos nomes dos meses. Estas linhas da disposição têm nove caracteres de extensão, seja qual for o número de letras em cada um deles. A forma mais fácil de estabelecer uma comparação efectiva é dimensionar a inserção da sequência igualmente para nove caracteres.

Ensaio do módulo 2.1.1

Precisaremos de inserir em modo directo O$, P1, P2 e P$. Agora, GOTO (passar) ao módulo e ele deverá imprimir o aviso P$ na posição especificada e convidá-lo a confirmar a sua inserção.

MÓDULO 2.1.2

```
1000 REM ************************
         THE FIRST TIME THIS
         PROGRAM IS RUN IT WILL
         BE NECESSARY TO ENTER
         IN DIRECT MODE:
         "LET MO=X"
         WHERE X IS THE NUMBER
         OF THE CURRENT MONTH.
1010 REM ************************
1020 LET H=1
1030 DIM O$(32)
1035 LET K$="_____________________
_____________"
1040 LET L$="********************
************"
1050 DATA "JANUARY","FEBRUARY","
MARCH","APRIL","MAY","JUNE","JUL
Y","AUGUST","SEPTEMBER","OCTOBER
","NOVEMBER","DECEMBER"
1060 DIM N$(12,9)
 1070 RESTORE
 1080 FOR I=1 TO 12
 1090 READ N$(I)
 1100 NEXT I
 1180 LET P2=0
 1185 CLS
```

Este módulo organiza as necessárias variáveis. Saliente-se o uso da função DATA (dado) e dos comandos que lhe estão associados, READ e RESTORE (ler e restabelecer).

Comentário

Linhas 1000-1010. Uma vez inserido MO pela primeira vez, é sensato deixar este enunciado REM no lugar, para o caso de, futuramente, passar o programa inadvertidamente e apagar MO.

Linha 1020. Esta variável será explicada no módulo 5.

Linhas 1030-1040. Estas sequências são utilizadas para dar formato ao *écran*.

Linhas 1050-1100. A disposição N$ é preenchida com os nomes dos meses, utilizando READ, DATA e RESTORE. Salienta-se que RESTORE é necessário porque este programa começa sempre com GOTO 1, o que não recoloca os indicadores de READ no início dos dados. Se o programa for passado uma segunda vez, o indicador terá já chegado ao fim dos dados e você obterá uma informação de erro por insuficiência de dados.

Ensaio do módulo 2.1.2

Este é mais indicado para quando tiverem sido inseridos os outros módulos que utilizam as variáveis.

MÓDULO 2.1.3

```
3590 REM ************************
3600 REM REGISTER MONTH
3610 REM ************************
3615 LET Y=MO+11
3620 PRINT AT 0,10;"HOME BUDGET"
3630 LET P$="WHAT MONTH IS IT?"
3650 LET P1=3
3660 GO SUB 3360
3670 IF T$=N$(MO) THEN RETURN
```

```
3680 FOR I=1 TO 12
3690 IF T$=N$(I) THEN GO TO 3740
3700 NEXT I
3710 CLS
3720 PRINT AT 9,0;"THERE MUST BE
 A MISTAKE IN THE  MONTH YOU GAV
E. I DON'T KNOW OF A MONTH CALLE
D ";Q$;"!"
3730 GO TO 3620
3740 LET M2=I
3750 FOR I=M0 TO M2-1+12*(M2<M0)
3755 LET I1=I-12*(I>12)
3760 CLS
3770 PRINT AT 0,11;"UPDATE"
3780 PRINT '''"PLEASE INPUT AMOUN
TS FOR NEXT"'N$(I)
3790 FOR J=1 TO N1
3800 LET P1=5
3830 LET P$=B$(1,J)+":("+STR$ B(
1,J,I1)+"):"
3840 GO SUB 3360
3850 LET B(1,J,I1)=VAL Q$
3860 PRINT AT 5,0;O$
3870 NEXT J
3900 LET P$="MAIN INCOME:("+STR$
 E(1,I1)+"):"
3910 GO SUB 3360
3920 LET E(1,I1)=VAL Q$
3950 LET P$="ADDITIONAL INCOME:(
"+STR$ E(3,I1)+"):"
3960 GO SUB 3360
3970 LET E(3,I1)=VAL Q$
3980 NEXT I
3990 LET M0=M2
4000 LET Y=M0+11
4010 GO SUB 2600
4015 GO SUB 2900
4020 RETURN
```


O objectivo deste módulo é verificar se o mês mudou desde a última utilização do programa. Cado isso tenha acontecido, são apagados os dados desactualizados e pedidos novos dados. Se a mudança for de Maio para Junho, os dados relativos a Maio serão apagados, sendo introduzidos novos dados para o mês de Maio seguinte. Deste modo, o período tratado pelo programa começa sempre pelo mês corrente e cobre um período de doze meses.

Comentário

Linha 3615. Y é o número do mês que finaliza o período de doze meses.

Linhas 3630-3660. Este é um exemplo de como o módulo 1 é solicitado.

Linhas 3670-3730. O nome do mês é comparado com o mês corrente do programa. Se não forem o mesmo, o módulo verifica o número do mês ou emite uma mensagem de erro.

Linhas 3740-4020. Esta secção do programa não pode ser ensaiada enquanto não tiverem entrado dados actuais, embora o seu objectivo geral seja aceitar dados actualizadores de informações, caso o mês tenha mudado desde a última utilização.

Linha 3750. MO é o número do último mês em que o programa foi usado. M2 é o número do mês corrente.

Linha 3790. N1 é o número total de denominações de pagamento previamente introduzidas.

Linhas 3800-3860. São pedidos novos números para cada denominação de pagamento, uma de cada vez. Os números referentes ao mês prestes a ser apagado são mostrados, como forma de guia.

Linhas 3900-3970. Repete-se o procedimento para os rendimentos principais e adicionais.

Linha 4010. A sub-rotina solicitada por esta linha volta a calcular a análise orçamental.

Ensaio do módulo 2.1.3

Apenas poderá ensaiar se o módulo é capaz de lidar com as mudanças dos meses e reconhecer nomes de meses inválidos. Para o fazer, após ter inserido MO em modo directo, comece a passar o programa com GOTO 1 e experimente inserir alguns nomes de meses inválidos, os quais deverão ser rejeitados. Ao inserir o mês correcto, dar-se-á o regresso do módulo — precisará de introduzir um comando temporário de STOP na linha 3589, para evitar que ele simplesmente volte a executar o módulo. Experimente também inserir o mês a seguir ao corrente... O programa deverá parar, declarando erro 2, linha 3790.

MÓDULO 2.1.4

```
1510 REM ************************
1520 REM SET UP REGULAR PAYMENTS
1530 REM ************************
1540 DIM B$(2,20,9)
1550 DIM C(20)
1560 DIM B(2,20,12)
1570 DIM D(4,12)
1580 DIM E(4,12)
1590 DIM F(2,12)
1620 LET N1=0
1630 LET N2=0
1635 DIM T(2)
1640 GO SUB 4030
1650 GO SUB 3050
1660 GO SUB 2900
1670 RETURN
```

Este módulo organiza as várias disposições utilizadas para armazenar os dados e pede as inserções iniciais naquelas disposições. De notar que a solicitação deste módulo resulta na perda de quaisquer dados armazenados.

Comentário

Linha 1540. B$ é a disposição em que armazenaremos os nomes das denominações de pagamento: tomam-se providências para vinte denominações. A primeira dimensão da disposição é 2, pois o programa providencia dois conjuntos paralelos de dados, um deles com os números reais e o outro como cópia em que poderá fazer alterações hipotéticas sem afectar os verdadeiros dados.

Linha 1550. C é a disposição onde ficará, para cada denominação de pagamento, o pagamento médio mensal para satisfazer os compromissos de um ano.

Linha 1560. A disposição B guardará os reais pagamentos mensais para os dados reais e hipotéticos.

Linha 1570. A disposição D guardará o total de pagamentos mensais para cada mês. Guardará ainda uma determinação sobre se o orçamento médio mensal cobriu, de facto, a quantia a ser paga nesse mês.

Linha 1580. A disposição E guarda pormenores mensais de rendimentos reais e hipotéticos.

Linha 1590. A disposição F guarda o balanço de rendimentos sobre despesas.

Linhas 1620-1630. N1 e N2 referem-se ao número de denominações de pagamento registados nos lados real e hipotético dos dados.

Linhas 1640-1660. Tendo organizado as disposições, o utilizador é solicitado a fazer uma introdução inicial de rendimentos e pagamentos, os quais são depois copiados para o lado hipotético da disposição pela sub-rotina na linha 2900.

Ensaio do módulo 2.1.4

O ensaio deste módulo deve esperar até que os outros módulos façam uso das disposições.

```
1190 GO SUB 3590
1210 PAPER 7: INK 0: CLS : PRINT
 "       HOME BUDGET"
1220 PRINT '"FUNCTIONS AVAILABLE
;"
1230 PRINT PAPER 5;"1)INITIALISE
"
1240 PRINT "2)RESET HYPOTHETICAL
 BUDGET"
1250 PRINT PAPER 5;"3)DISPLAY MO
NTHLY ANALYSIS"
1260 PRINT "4)HYPOTHETICAL ANALY
SIS"
1270 PRINT PAPER 5;"5)REAL CHANG
ES"
1280 PRINT "6)HYPOTHETICAL CHANG
ES"
1290 PRINT PAPER 5;"7)NEW BUDGET
 HEADINGS"
1300 PRINT "8)HYPOTHETICAL BUDGE
T HEADINGS"
1302 PRINT PAPER 5;"9)DELETE REA
L BUDGET HEADING"
1304 PRINT "10)DELETE HYPOTHETIC
AL HEADING"
1310 PRINT PAPER 5;"11)STOP"
1330 LET P1=15
1340 LET P$="WHICH DO YOU REQUIR
E?"
1350 GO SUB 3360
1360 CLS
1370 LET Z$=Q$
1380 LET H=1+(Z$="2")+(Z$="4")+(
Z$="6")+(Z$="8")+(Z$="10")
1390 IF Z$="1" THEN GO SUB 1520
1400 LET N=N1
1410 IF H=2 THEN LET N=N2
```



```
1420 IF Z$="2" THEN GO SUB 2900
1430 IF Z$="3" OR Z$="4" THEN GO
 SUB 1690
1440 IF Z$="5" OR Z$="6" THEN GO
 SUB 4290
1450 IF Z$="7" OR Z$="8" THEN GO
 SUB 3050
1452 IF Z$="9" OR Z$="10" THEN G
O SUB 4870
1460 IF Z$="11" THEN GO TO 1490
1470 CLS
1480 GO TO 1200
1490 PRINT FLASH 1;AT 10,0;"DO Y
OU WISH TO RE-RECORD? (Y/N)": IN
PUT Q$: CLS : IF Q$="Y" THEN SAV
E "BUDGET": BEEP 1,2: PRINT AT 1
0,0;"REWIND, THEN ANY KEY TO VER
IFY": PAUSE 0: VERIFY "BUDGET":
STOP
1500 STOP
```

Trata-se de um módulo de ementa padrão.

Comentário

Linha 1380. Não existem secções separadas do programa para tratar da manipulação dos dados hipotéticos. Cada módulo programático trabalha sobre os dados reais, se H = 1, e sobre os dados hipotéticos, se H = 2. Esta linha usa condições lógicas como variáveis.

Linha 1390. Dado que não há razão para existir o mesmo número de denominações de pagamento nas secções real e hipotética dos dados, coloca-se a variável N igual a N1 ou N2, dependendo de H ser igual a 1 ou 2.

Ensaio do módulo 2.1.5

Introduza a linha 3320 e carregue em RETURN (regresso). O programa deverá agora aceitar a inserção no módulo de todas as funções, excepto a função 1. Nada acontecerá, é claro, a não ser o funcionamento da função de *stop*.

MÓDULO 2.1.6

```
3040 REM **********************
3050 REM INPUT OF PAYMENTS
3060 REM **********************
3070 PRINT "      INPUT OF BILLS"
3080 PRINT '''"PRECEDE NAME OF IT
EM WITH A * IFYOU DO NOT WANT IT
 BUDGETED."
3090 LET P$="HEADING (""ZZZ"" TO
 QUIT):"
3100 LET P1=6
3110 GO SUB 3360
3115 IF Q$="ZZZ" THEN GO SUB 259
0: RETURN
3130 PRINT AT 6,0;P$;Q$
3140 IF H=1 THEN LET N1=N1+1
3150 IF H=2 THEN LET N2=N2+1
3160 LET N=N1*(H=1)+N2*(H=2)
3170 IF N<=20 THEN GO TO 3210
3180 PRINT AT 8,0;"NO MORE ROOM
IN PAYMENTS FILE."
3190 PRINT '''" Press any key to
 continue."
3195 PAUSE 0
3200 RETURN
3210 LET B$(H,N)=Q$
3220 FOR I=MO TO Y
```



```
3230 LET I1=I-12*(I>12)
3240 LET P$="AMOUNT FOR "+N$(I1)
+":"
3250 LET P1=I+7-MO
3260 GO SUB 3360
3270 LET B(H,N,I1)=VAL Q$
3280 PRINT AT I+7-MO,22;Q$
3290 NEXT I
3300 CLS
3320 GO TO 3070
```

O objectivo do módulo consiste em aceitar novas denominações de pagamento e pedir pormenores de pagamentos sob as denominações durante um período de doze meses.

Comentário

Linha 3080. Assuntos precedidos de um asterisco não serão incluídos no cálculo do orçamento médio mensal.

Linha 3160. Note-se o modo como são usadas as condições lógicas para incrementar N1 ou N2.

Linha 3120. Note-se o modo como o valor de H é utilizado para determinar em que metade da disposição estão colocados os dados.

Linha 3220. Se já passou o fim do ano civil, a I1 é subtraído 12, por forma a permanecer dentro dos limites 1 a 12.

Linha 3250. O valor do desvio é utilizado para formatar a impressão de sucessivas P$.

Ensaio do módulo 2.1.6

Uma vez iniciado, o programa deverá aceitar denominações de pagamento em conjunto com pagamentos associados, armazenan-

do-os, quer do lado real, quer do lado hipotético das disposições atinentes. Isto apenas pode ser verificado pela impressão de elementos das disposições B$ e B, em modo directo.

MÓDULO 2.1.7

```
4030 REM ************************
4040 REM INCOME
4050 REM ************************
4060 PRINT "                INCOME"
4070 PRINT '''"PLEASE INPUT SALAR
Y FOR 12      MONTHS AS FOLLOWS:
"
4080 FOR I=MO TO Y
4090 LET I1=I-12*(I>12)
4100 LET P$=N$(I1)+":"
4110 LET P1=I+5-MO
4120 GO SUB 3340
4130 LET E(H,I1)=VAL Q$
4140 PRINT AT I+5-MO,10;E(H,I1)
4150 NEXT I
4160 CLS
4170 PRINT "NOW INPUT ANY OTHER
ANTICIPATED INCOME:"
4180 FOR I=MO TO Y
4190 LET I1=I-12*(I>12)
4200 LET P1=I+5-MO
4210 LET P$=N$(I1)+":"
4220 GO SUB 3360
4230 LET E(H+2,I1)=VAL Q$
4240 PRINT AT I+5-MO,10;E(H+2,I1
)
4250 NEXT I
4260 CLS
4270 RETURN
```


Este método aceita dados sobre rendimentos durante os doze meses, sob as denominações de salários e de outros rendimentos antecipados.

Comentário

Linhas 4080-4150. Este desvio aceita as quantias de salários a serem colocadas na disposição E.

Linhas 4170-4250. O mesmo que nas anteriores, embora para rendimentos suplementares.

Ensaio do módulo 2.1.7

Tem-se acesso a este módulo via módulo de iniciais, o qual deverá solicitar a partir da ementa. Ser-lhe-á pedido que introduza denominações de pagamento e depois, ao terminar ZZZ, que introduza rendimentos sob as duas denominações. O programa deverá regressar à ementa, após as quantias referentes ao rendimento terem sido introduzidas.

MÓDULO 2.1.8

```
3500 REM ************************
3510 REM MONEY$
3520 REM ************************
3530 LET M=M+1000.0001*(M>=0)-10
00.0001*(M<0)
3540 LET M$=(STR$ ABS M)(2 TO 4)
3550 IF ABS M>=2000 THEN LET M$=
"***"
3560 IF M<0 THEN LET M$=""+M$+""
3570 RETURN
```

Este módulo constrói sequências a partir dos dados armazenados que representem quantias em numerário. Isto é feito com a intenção de tornar mais fácil a formatação dos números.

Comentário

Linha 3530. Acrescenta-se 1000.0001 para assegurar que haja três algarismos antes do ponto decimal e dois após ele, sem alterar o valor de quaisquer algarismos que possam já encontrar-se nessas posições. M é uma variável temporária e o processo não afecta o valor do número original.

Linha 3550. Ao apresentar os dados, o programa apenas pode mostrar os três dígitos antes do ponto decimal. Se isto for um sério revés para si, é simples alterá-lo, de forma a que ele possa mostrar valores até 9999. Esta limitação não afecta aquilo que está armazenado, tão-só aquilo que é mostrado.

Linha 3560. O programa precisa de ter a capacidade de mostrar números negativos, tornando claro que são negativos. Isto faz-se através da impressão deles em vídeo invertido. Não é visível na impressão no *écran*, mas a sequência criada no final da linha consiste em:

«[4 com CAPS SHIFT]» + M$ + «[3 com CAPS SHIFT]»

As duas sequências aparentemente vazias que são colocadas em ambas as extremidades de M$ acrescentam dois caracteres de controlo. Um deles coloca aquilo que se segue em vídeo invertido. O outro regressa ao vídeo normal.

Ensaio do módulo 2.1.8

Em modo directo, introduzir LET M = 1 e a seguir GOSUB 3500. Depois, imprima a M$ criada. Experimente isto com outros valores, incluindo alguns números negativos.


```
2590 REM ************************
2600 REM UPDATE BUDGET
2610 REM ************************
2620 LET T(H)=0
2630 FOR I=1 TO N
2640 LET BUDGET=0
2650 IF B$(H,I,1)="*" THEN GO TO
 2710
2660 FOR J=1 TO 12
2670 LET BUDGET=BUDGET+B(H,I,J)
2680 NEXT J
2690 LET C(I)=BUDGET/12
2700 LET T(H)=T(H)+C(I)
2710 NEXT I
2720 LET TTOTAL=0
2730 LET CUM=0
2740 FOR I=MO TO Y
2750 LET I1=I-12*(I>12)
2760 LET D(H,I1)=0
2770 FOR J=1 TO N
2780 LET D(H,I1)=D(H,I1)+B(H,J,I
1)
2790 NEXT J
2800 LET TTOTAL=TTOTAL+D(H,I1)
2810 FOR J=1 TO N
2820 IF B$(H,J,1)="*" THEN LET T
TOTAL=TTOTAL-B(H,J,I1)
2830 NEXT J
2840 LET D(H+2,I1)=T(H)*(I-MO+1)
-TTOTAL
2850 LET CUM=CUM+E(H,I1)+E(H+2,I
1)-D(H,I1)
2860 LET F(H,I1)=CUM
2870 NEXT I
2880 RETURN
```

Este módulo executa os cálculos que fornecem a variedade de análises de rendimentos e despesas. Na verdade, é bastante menos complexo do que parece, dado que apenas estão envolvidas simples adições, subtracções e divisões por 12.

Comentário

Linha 2620. T é uma disposição de dois elementos que é utilizada para armazenar a soma dos pagamentos individuais mensais do orçamento, com o objectivo de fornecer um orçamento geral mensal.

Linhas 2630-2710. Neste desvio, o total de pagamentos anuais para cada denominação de pagamento é encontrado e dividido por 12, para dar uma média mensal. A disposição C armazena a média mensal para cada denominação de pagamento. Saliente-se que nomes de pagamentos começados por «*» estão isentos deste procedimento.

Linha 2740-2790. A disposição D é preenchida com o total de pagamentos em cada mês.

Linhas 2800-2840. O total de pagamentos de cada mês é armazenado na disposição D. Estes totais são acumulados em TTOTAL e de TTOTAL são subtraídos quaisquer pagamentos que o utilizador não queira ver incluídos no orçamento médio. Por cada passagem pelo desvio, começando em 2740, TTOTAL é determinado para conter o total de pagamentos, até e incluindo o mês I1, que pertencem a denominações de pagamento incluídas no orçamento mensal médio. Os números resultantes constituem o balanço do orçamento mensal até àquele mês que estiver acima do total de pagamentos de assuntos previstos no orçamento. Isto é armazenado na segunda metade da disposição D. E, se tudo isto lhe soar tão confuso como a mim, tenha fé e espere até ter a oportunidade de passar alguns números simples, voltando então a ler o que foi aqui escrito.

Linha 2850. O balanço de rendimentos sobre despesas é armazenado como CUM, sob a forma de elementos sucessivos da disposição F.

Ensaio do módulo 2.1.9

Este módulo será ensaiado após a introdução do próximo.

MÓDULO 2.1.10

```
1680 REM ************************
1690 REM DISPLAY BILLS
1700 REM ************************
1710 PRINT "          PAYMENTS"
1720 LET P$="MONTH TO START"
1730 LET P1=3
1740 GO SUB 3360
1750 FOR I=1 TO 12
1760 IF T$=N$(I) THEN GO TO 1790
1770 NEXT I
1780 GO TO 1720
1790 LET M1=I
1800 IF M0-M1+12*(M1>M0-1)<4 THE
N LET M1=M0-4+12*(M0<5)
1810 CLS
1820 PRINT "MONTH     —   —   —
  —   —  B  —"
1830 OVER 1: PRINT AT 0,0;TAB 10
;N$(M1, TO 3);TAB 14;N$(M1+1-12*
(M1>11), TO 3);TAB 18;N$(M1+2-12
*(M1>10), TO 3);TAB 22;N$(M1+3-1
2*(M1>9), TO 3)'L$: OVER 0
1840 FOR I=1 TO N
1843 PAPER 5: IF I/2<>INT (I/2)
THEN PAPER 7
1850 IF I<>11 THEN GO TO 1920
1860 PRINT AT 20,0;"PRESS ANY KE
Y TO CLS & CONTINUE"
1870 PAUSE 0
1880 FOR J=2 TO 21
1890 PRINT AT J,0;O$
1900 NEXT J
```

```
1910 PRINT AT 2,0;
1920 PRINT B$(H,I);"_";
1930 FOR J=M1 TO M1+3
1940 LET M=B(H,I,J-12*(J>12))
1950 GO SUB 3500
1960 PRINT M$;"_";
1970 NEXT J
1980 LET M=C(I)
1990 GO SUB 3500
2000 PRINT M$;"_"
2010 GO SUB 3500
2020 NEXT I
2030 PRINT K$
2040 PRINT '''"PRESS ANY KEY FOR
ANALYSIS"
2050 PAUSE 0
2070 FOR I=2 TO 21
2080 PRINT AT I,0;O$
2090 NEXT I
2110 PAPER 5: PRINT AT 2,0;"TOTA
L     _"''"BUDG.BAL._"''"OTHER IN
C_"''"CASH BAL._"
2115 PAPER 7: PRINT AT 3,0;"BUDG
ET    _"''"PAY      _"''"TOTAL IN
C_"''"CUM. BAL._"
2130 FOR I=M1 TO M1+3
2140 OVER 1: PRINT AT 2,0;
2150 LET I1=I-12*(I>12)
2160 LET I2=10+4*(I-M1)
2170 LET M=D(H,I1)
2180 GO SUB 3500
2190 PAPER 5: PRINT TAB I2;M$;"__
"
2200 LET M=T(H)
2210 GO SUB 3500
2220 PAPER 7: PRINT TAB I2;M$;"__
"
2230 LET M=D(H+2,I1)
2240 GO SUB 3500
```



```
2280 PAPER 5: PRINT TAB I2;M$;"__
"
2290 LET M=E(H,I1)
2300 GO SUB 3500
2310 PAPER 7: PRINT TAB I2;M$;"__
"
2320 LET M=E(H+2,I1)
2330 GO SUB 3500
2340 PAPER 5: PRINT TAB I2;M$;"__
"
2350 LET M=E(H,I1)+E(H+2,I1)
2360 GO SUB 3500
2370 PAPER 7: PRINT TAB I2;M$;"__
"
2380 LET M=(E(H,I1)+E(H+2,I1)-D(
H,I1))
2400 GO SUB 3500
2440 PAPER 5: PRINT TAB I2;M$;"__
"
2450 LET M=F(H,I1)
2460 GO SUB 3500
2500 PAPER 7: PRINT TAB I2;M$;"__
"
2510 NEXT I
2520 OVER 0: PRINT K$( TO 20)
2530 LET P$="DO YOU WISH TO SEE
FIGURES      AGAIN? (Y/N):"
2540 LET P1=20
2550 GO SUB 3360
2560 IF Q$<>"Y" THEN RETURN
2570 CLS
2580 GO TO 1810
```

O objectivo deste módulo, de aspecto tão assustador, é passar (mostrar) os vários dados e análises sob uma forma tabular. De particular interesse é o modo como a tabela é formada, em grande parte, com base em variáveis já activas dentro do programa — se os dados tiverem uma estrutura regular poderão ditar o formato da sua própria apresentação sem grandes dificuldades.

Comentário

Linhas 1700-1810. Esta secção aceita a inserção de um mês de partida para a tabela, experimentando a sua validade. O mês de partida nunca poderá ser posterior ao 9.° do período de doze meses, dado que são passados quatro meses.

Linha 1820-1830. Estas duas linhas imprimem a denominação da tabela, incluindo as três primeiras letras do nome do mês.

Linhas 1840-2020. Este desvio imprime as denominações de pagamento e os pagamentos mensais associados àqueles durante o período de quatro meses. Para fazer destacar as linhas separadas da tabela, faz-se variar a cor do papel, dependendo de a variável do desvio ser ímpar ou par.

Linhas 2060-2090. Utilizando uma linha de espaços vazios e um desvio, limpa-se com eficácia parte do *écran*.

Linhas 2110-2510. Esta secção imprime as várias análises de rendimentos e despesas. Note-se a forma como a variável do desvio é usada (2510, 2610) para criar duas variáveis mais, sobre as quais se baseia a formatação dos dados. Depois da impressão das denominações, nas linhas 2110-2120, o exame das linhas seguintes mostra que elas se dividem em oito secções. Cada secção começa com a organização da variável temporária M e com a solicitação do módulo 8, seguida da impressão de M$.

Ensaio do módulo 2.1.10

Apague qualquer dado já armazenado, volte a introduzir MO em modo directo e passe o programa, com GOTO 1. Introduza rendimentos e alguns pagamentos, mas mantenha-os simples — dezenas ou centenas, em números inteiros. Solicite o módulo 3 e examine o resultado. Se houver problema com a formatação, então talvez tenha cometido um erro na inserção do módulo 10. Se os números não fizerem sentido, o erro poderá ser no módulo 9. Se os pagamentos reais estiverem errados, verifique a entrada de pagamentos, no módulo 6. Para averiguar da causa de erros nos números dos rendimentos, inspeccione o módulo 7.


```
2890 REM **********************
2900 REM SET UP SHADOW ARRAY
2910 REM **********************
2920 FOR I=1 TO N1
2930 LET B$(2,I)=B$(1,I)
2940 FOR J=1 TO 12
2950 LET B(2,I,J)=B(1,I,J)
2960 LET D(2,J)=D(1,J)
2970 LET D(4,J)=D(3,J)
2980 LET E(2,J)=E(1,J)
2990 LET E(4,J)=E(3,J)
2995 LET F(2,J)=F(1,J)
3000 NEXT J
3010 NEXT I
3020 LET N2=N1
3030 RETURN
```

Este módulo copia os dados da metade real da disposição para a metade hipotética. Se houvesse alguma diferença entre as metades real e hipotética, ela perder-se-ia, caso esta função fosse utilizada.

Ensaio do módulo 2.1.11

Está agora em posição de ensaiar uma separação correcta entre as áreas de dados reais e hipotéticos. Introduza alguns novos pagamentos hipotéticos, verifique que eles foram aceites solicitando a função 4 da ementa, voltando depois a esta e solicitando a função 3. A tabela mostrada não deve apresentar indícios dos pagamentos hipotéticos que introduziu.

```
4280 REM ************************
4290 REM CHANGES
4300 REM ************************
4310 PRINT "                CHANGES
"
4320 PRINT ''"1)CHANGE EXISTING
BUDGET HEAD"'"2)CHANGE MAIN INCO
ME"'"3)CHANGE ADDITIONAL INCOME"
4340 LET P$="WHICH DO YOU REQUIR
E?"
4350 LET P1=7
4360 GO SUB 3360
4370 LET H$=Q$
4380 CLS
4410 IF H$="1" THEN GO SUB 4460
4420 IF H$="2" OR H$="3" THEN GO
 SUB 4690
4440 GO SUB 2600
4450 RETURN
4460 LET P$="NAME OF BUDGETARY H
EADING TO BE CHANGED ?"
4470 LET P1=1
4480 GO SUB 3340
4490 FOR I=1 TO N
4500 IF T$=B$(H,I) THEN GO TO 45
60
4510 NEXT I
4520 PRINT ''" NO HEADING OF THA
T NAME."''" Press any key to co
ntinue."
4530 PAUSE 0
4550 RETURN
4560 LET B=I
4570 LET P$="NEW FIGURE OR ""Z""
 TO LEAVE."
```



```
4580 LET P1=17
4590 FOR I=M0 TO Y
4600 LET I1=I-12*(I>12)
4610 PRINT AT I-M0+3,0;N$(I1);":
";B(H,B,I1)
4620 GO SUB 3340
4630 IF Q$="Z" THEN GO TO 4660
4640 LET B(H,B,I1)=VAL Q$
4650 PRINT AT I-M0+3,15;"(";B(H,
B,I1);")"
4660 NEXT I
4670 GO SUB 2590
4680 RETURN
4730 LET X=2*(H$="3")
4750 PRINT "MAIN INCOME:" AND (H
$="2");"SUPPLEMENTARY INCOME:" A
ND (H$="3")
4760 LET P$="INPUT NEW FIGURE OR
 ""Z"" TO LEAVE"
4770 LET P1=17
4780 FOR I=M0 TO Y
4790 LET I1=I-12*(I>12)
4800 PRINT AT I+3-M0,0;N$(I1);":
";E(H+X,I1)
4810 GO SUB 3340
4820 IF Q$="Z" THEN GO TO 4850
4830 LET E(H+X,I1)=VAL Q$
4840 PRINT AT I-M0+3,15;"(";E(H+
X,I1);")"
4850 NEXT I
4860 RETURN
```

Este módulo permite que se façam alterações a dados existentes.

Comentário

Linhas 4460-4680. São passados pagamentos mensais correntes, sob uma denominação de pagamento específica, podendo ser confirmados ou alterados. É então recalculada a análise orçamental.

Linhas 4730-4860. É organizada uma variável X, dependendo de terem sido especificados rendimentos principais ou adicionais. Os números introduzidos são armazenados na secção relevante da disposição E, de acordo com o valor de X.

Ensaio do módulo 2.1.12

Solicite o módulo e faça algumas alterações aos dados existentes, tanto reais como hipotéticos.

MÓDULO 2.1.13

```
4870 REM **********************
4880 REM DELETE BUDGET HEAD
4890 REM **********************
4900 LET P$="NAME OF ITEM TO BE
DELETED?"
4910 LET P1=1
4920 GO SUB 3340
4930 FOR I=1 TO N
4940 IF T$=B$(H,I) THEN GO TO 50
00
4950 NEXT I
4960 PRINT AT 2,8;"ITEM NOT FOUN
D.""/" Press any key to continue
."
```



```
4970 PAUSE 0
4980 RETURN
5000 IF H=1 THEN LET N1=N-1
5010 IF H=2 THEN LET N2=N-1
5015 LET N=N1*(H=1)+N2*(H=2)
5020 FOR J=I TO N
5030 LET B$(H,J)=B$(H,J+1)
5040 FOR K=1 TO 12
5050 LET B(H,J,K)=B(H,J+1,K)
5060 NEXT K
5070 NEXT J
5120 GO SUB 2600
5130 RETURN
```

Este módulo define uma denominação de pagamento existente, na área real ou hipotética, juntamente com os respectivos pagamentos associados.

Comentário

Linhas 4930-4980. O nome do assunto a ser apagado é comparado com denominações de pagamento já existentes.

Linhas 5000-5120. N1 ou N2 são decrescidas se forem apagados dados, respectivamente, dos lados real ou hipotético. Os assuntos entre B$ e B são então deslocados para baixo, um de cada vez, para que se possa reescrever na posição dos dados a serem apagados. É recalculada a análise orçamental.

Ensaio do módulo 2.1.13

Para fazer o ensaio basta apagar algumas denominações. Se esta operação resultar, o programa estará correctamente introduzido e pronto a ser utilizado.

Sumário

Este longo programa é uma ferramenta poderosa, se correctamente utilizada, embora requeira alguma prática até que se consiga tirar dele o máximo rendimento. Se for tomado a sério, poderá dar-lhe algumas informações surpreendentes quanto ao estado das suas finanças para o ano que se avizinha: quando se tornará necessário apertar o cinto e quando haverá alguma abundância; como poderão reordenar-se os pagamentos, de modo a assegurar alguns extras para o Natal ou para as férias, e quais serão as consequências gerais de um novo encargo ou de um novo acréscimo nos rendimentos.

Isto não é, contudo, o fim da questão. Sem grandes adaptações, o enquadramento do programa poderá ser aplicado a uma variedade de utilizações, onde os dados devem ser mostrados e analisados dentro de várias denominações, onde será preciso fazer alterações à vontade, onde deverá existir a possibilidade de fazer acessos hipotéticos ou mudanças, sem corromper os dados existentes.

Com alterações ao módulo 9, pode facilmente conseguir-se que o programa realize cálculos muito diferentes dos aqui realizados. Lembre-se de que o mais difícil de pôr a funcionar não são as linhas *Basic* individuais, quando se junta uma nova aplicação... antes pô-las a funcionar em conjunto. Logo que este programa esteja a funcionar a seu contento, incluindo quaisquer melhoramentos que queira fazer-lhe, olhe-o como um enquadramento para as suas ideias e tarefas, de preferência a algo que apenas poderá ser aplicado ao seu orçamento familiar.

Adicionalmente, deverá ter ganho alguma confiança, ao introduzir este programa, quanto à técnica de tratamento e subdivisão de vastas disposições, pelo simples uso das variáveis do programa. Tem exemplos práticos em relação ao modo como variáveis de desvios e outras podem ser utilizadas para ajudar a formatar os dados.

Por último, tem um exemplo de um módulo simples para a formatação e verificação de avisos programáticos, uma vantagem potencial em programas interactivos.

Avançando mais

1) Tal como o «Unificha», este programa está escrito com quase nenhuma utilização de linhas multienunciado. Encurte o programa o mais que puder pela combinação de linhas individuais.

2) Excepto onde se revelou absolutamente necessário à clareza de apresentação, não foi feita qualquer tentativa de tornar interactiva a impressão do programa, em termos de cor. Percorra o programa e torne-o mais alegre, colocando alguns comandos de cor nos sítios certos.

3) Dê uma vista de olhos pelas disposições definidas no módulo 4. Várias delas podem facilmente ser combinadas. Isto poderá resultar em grande economia ao imprimir a tabela de análises, visto que o módulo poderá apenas funcionar metodicamente através de uma disposição, em vez de o fazer em quatro ou cinco. Experimente e veja.

2.2 CONTABILISTA

Este programa simples permite ao utilizador criar facilmente contas claras e compreensíveis. Não é levada a cabo qualquer análise, a não ser para gerar subtotais e totais onde forem indicados. Assim, se os livros não condisserem, terá de ser você a esclarecer o assunto. Uma vez que o programa é bastante simples, os comentários sobre os módulos serão bastante mais breves do que antes.

MÓDULO 2.2.1

```
1000 REM **********************
1010 REM MENU
1020 REM **********************
1030 PAPER 7: INK 0: CLS : PRINT
 "          ACCOUNTS"
1040 PRINT '"FUNCTIONS AVAILABLE
:"
```

```
1050 PRINT '"1)INPUT NEW HEADING
S"
1060 PRINT '"2)CHANGE AMOUNTS/DE
LETE ITEMS"
1070 PRINT '"3)PRINT ACCOUNTS"
1080 PRINT '"4)INITIALISE ACCOUN
TS"
1090 PRINT '"5)STOP"
1100 INPUT Z$
1110 CLS
1120 IF Z$<>"4" AND Z$<>"5" THEN
 GO SUB 1320
1130 CLS
1140 IF Z$="1" THEN GO SUB 1390
1150 IF Z$="2" THEN GO SUB 2120
1160 IF Z$="3" THEN GO SUB 2660
1170 IF Z$="4" THEN GO SUB 1210
1180 IF Z$="5" THEN GO TO 1202
1190 CLS
1200 GO TO 1000
1204 PRINT FLASH 1;AT 10,10;"ACC
OUNTANT": INPUT "Do you want to
 re-record? (Y/N)";Q$: IF Q$="Y"
THEN SAVE "ACCOUNTANT": BEEP 1,2
: PRINT AT 10,0;"REWIND, THEN AN
Y KEY TO VERIFY.": VERIFY "ACCOU
NTANT"
1208 STOP
```

Esta é uma ementa de programa padrão.

MÓDULO 2.2.2

```
1210 REM ************************
1220 REM CREDIT/DEBIT
1230 REM ************************
1240 DIM A$(2,100,15)
1250 DIM A(2,100)
1260 DIM C(2)
1270 LET C(1)=1
1280 LET C(2)=1
1290 DIM O$(32)
1300 LET L$="-------"
1310 RETURN
```

Este módulo analisa as variáveis envolvidas e resultará na perda de quaisquer dados previamente armazenados.

Comentário

Linha 1240. A$ guardará os nomes das denominações em duas secções, uma para créditos e a outra para débitos.

Linha 1250. A disposição guardará os números correspondentes às denominações em A$.

Linhas 1260-1280. C guardará o número de assuntos armazenados nas áreas de crédito e de débito.

MÓDULO 2.2.3

```
1320 REM ************************
1330 REM CREDIT OR DEBIT?
1340 REM ************************
1350 PRINT AT 10,3;"DO YOU WANT:
1)CREDIT"
1360 PRINT "                2)DEB
IT"
```

```
1370 INPUT CD
1380 RETURN
```

Este módulo prepara a variável CD de acordo com o desejo do utilizador de lidar com a área de crédito ou de débito dos dados.

Ensaio do módulo 2.2.3

Qualquer função seleccionada da ementa deverá resultar na solicitação deste módulo.

MÓDULO 2.2.4

```
1390 REM ************************
1400 REM INPUT ITEMS
1410 REM ************************
1430 PRINT "              NEW ITEMS"
1440 PRINT '"IS THE ITEM:1)A SIN
GLE ITEM"
1450 PRINT "               2)A MAIN
 HEADING"
1460 PRINT "               3)A SUB-
HEADING"
1470 PRINT '"              ""0"" TO
 QUIT FUNCTION"
1480 INPUT TYPE
1490 IF TYPE=0 THEN RETURN
1500 IF TYPE=3 THEN GO TO 1750
```


Este módulo é utilizado para especificar o tipo de denominação prestes a ser armazenada nas contas. Existem dois tipos:

1) Itens (ou assuntos) simples, que serão introduzidos directamente na coluna principal das contas.

2) Denominações principais, que não têm números adjuntos mas que agem como indicadores para grupos de:

3) Subdenominações, que serão identificadas e subtotalizadas em separado da coluna principal de números.

MÓDULO 2.2.5

```
1510 REM ************************
1520 REM SINGLE ITEM OR MAIN
     HEADING
1530 REM ************************
1540 LET Q=0
1550 PRINT AT 8,0;"NAME OF ITEM
?";
1560 INPUT Q$
1570 PRINT Q$
1580 IF TYPE=2 THEN GO TO 1620
1590 PRINT '"AMOUNT FOR ITEM? ";
1600 INPUT Q
1610 PRINT Q
1620 INPUT "Is this correct? (Y/
N)";R$
1640 IF R$="Y" THEN GO TO 1690
1650 FOR I=8 TO 14
1660 PRINT AT I,0;O$
1670 NEXT I
1680 GO TO 1550
1690 LET Q$=" "+Q$
1700 IF TYPE=2 THEN LET Q$(1)="*
"
1710 LET A$(CD,C(CD))=Q$
1720 LET A(CD,C(CD))=Q
1730 LET C(CD)=C(CD)+1
1735 CLS
1740 GO TO 1410
```

Este módulo aceita a inserção de denominações principais ou de assuntos simples, tal como foi definido antes.

Comentário

Linhas 1540-1700. O nome de um assunto é aceite conjuntamente com uma quantia, se a variável TYPE estiver em 1... indicando um único assunto. Se TYPE estiver em 2, então é acrescentado «*» em frente à denominação que serve de indicador e que seja uma denominação principal.

Linhas 1710-1740. Se o assunto não for uma denominação principal, a variável passa a conter a quantia referente ao assunto. Nesta secção, a denominação é armazenada em A$ e a quantia em A. A disposição deverá, é claro, ter sido preparada pelo módulo de iniciação.

MÓDULO 2.2.6

```
1750 REM ************************
1760 REM SUB-HEADING
1770 REM ************************
1780 PRINT '"NAME OF MAIN HEADIN
G? ";
1790 INPUT Q$
1800 PRINT Q$
1810 LET Q$="*"+Q$
1812 DIM T$(15)
1814 LET T$=Q$
1820 FOR I=1 TO C(CD)
1830 IF A$(CD,I)=T$ THEN GO TO 1
890
1840 NEXT I
1850 PRINT '"SORRY NO HEADING OF
 THAT NAME."
1860 PRINT ''"Press any key to c
```



```
ontinue.": PAUSE 0
1880 RETURN
1890 LET PLACE=I+1
1900 PRINT '"NAME OF SUB-HEADING
?";
1910 INPUT Q$
1920 PRINT Q$
1930 PRINT '"AMOUNT FOR SUB-HEAD
ING?";
1940 INPUT Q
1950 PRINT Q
1960 INPUT "Are these correct? (
Y/N)";R$
1980 IF R$="Y" THEN GO TO 2030
1990 FOR I=8 TO 16
2000 PRINT AT I,0;O$
2010 NEXT I
2015 PRINT AT 8,0;
2020 GO TO 1900
2030 LET Q$="£"+Q$
2040 FOR I=C(CD)+1 TO PLACE+1 ST
EP -1
2050 LET A$(CD,I)=A$(CD,I-1)
2060 LET A(CD,I)=A(CD,I-1)
2070 NEXT I
2080 LET A$(CD,PLACE)=Q$
2090 LET A(CD,PLACE)=Q
2100 LET C(CD)=C(CD)+1
2105 CLS
2110 GO TO 1410
```

Destina-se a aceitar subdenominações, tal como foi definido antes.

Comentário

Linhas 1750-1890. É introduzido o nome da denominação principal relevante, verificado em relação às denominações já armazenadas, e a posição da denominação principal relevante é armazenada na variável PLACE.

Linhas 1900-2030. São introduzidos e confirmados pelo utilizador o nome e a quantia da nova subdenominação.

Linhas 2040-2100. É criado um espaço, movendo todos os assuntos da posição PLACE um lugar para cima, dentro da disposição. A nova subdenominação é então colocada imediatamente após a denominação principal relevante.

Ensaio do módulo 2.2.6

Introduza uma subdenominação sob uma das denominações principais que introduziu ao ensaiar o último módulo. Verifique se a denominação e a quantia associada foram colocadas nas disposições relevantes.

MÓDULO 2.2.7

```
2980 REM ************************
2990 REM CREATE M$
3000 REM ************************
3005 LET M=M+.001
3010 LET M$=(STR$ M)( TO LEN STR
$ M-1)
3080 RETURN
```

Este módulo produz uma sequência padrão, com duas posições decimais a partir dos números ligados às várias denominações. De modo diferente ao do módulo similar, no anterior programa, não são acrescentados zeros no princípio do número.

MÓDULO 2.2.8

```
2660 REM ************************
2670 REM PRINT ACCOUNTS
2680 REM ************************
2690 PRINT AT 0,12;"CREDIT" AND
CD=1;AT 1,12;"DEBIT" AND CD=2
2700 LET TTOTAL=0
2710 LET STOTAL=0
2720 FOR I=1 TO C(CD)-1
2730 IF I>1 AND A$(CD,I,1)<>"£"
THEN PRINT
2740 LET TTOTAL=TTOTAL+A(CD,I)
2750 IF A$(CD,I,1)="£" THEN PRIN
T TAB 2;
2760 PRINT A$(CD,I,2 TO );
2770 IF A$(CD,I,1)="*" THEN GO T
O 2850
2780 IF A(CD,I)=0 THEN GO TO 285
0
2790 LET M=A(CD,I)
2800 GO SUB 2980
2810 PRINT TAB 25-LEN M$;
2820 IF A$(CD,I,1)<>"£" THEN PRI
NT TAB 32-LEN M$;
2830 IF A$(CD,I,1)="£" THEN LET
STOTAL=STOTAL+A(CD,I)
2840 PRINT M$
2850 IF STOTAL=0 OR A$(CD,I+1,1)
="£" THEN GO TO 2910
2860 PRINT TAB 18;L$
2870 LET M=STOTAL
2880 GO SUB 2980
2890 PRINT TAB 25-LEN M$;M$
2900 LET STOTAL=0
2910 NEXT I
2920 LET M=TTOTAL
2930 GO SUB 2980
```

```
2950 PRINT '"TOTAL:";TAB 32-LEN
M$;M$
2960 PRINT '"Press any key to co
ntinue."
2965 PAUSE 0
2970 RETURN
```

O objectivo deste módulo é imprimir e mostrar tanto o lado do crédito como o do débito, nas contas.

Comentário

Linha 2690. Note-se a utilização de AND para controlar aquilo que aqui está impresso. Quando dois assuntos são ligados por AND, adquirem o valor do primeiro assunto do par, se o segundo assunto do par tiver um valor positivo. Vimos já que condições tais como CD = 2 têm um valor de 1 ou 0, dependendo de serem verdadeiras ou falsas. Do mesmo modo, nesta linha, se CD = 1, então a expressão «CREDIT» AND CD = 1 adquire o valor «CREDIT». Se CD não for 1, então a expressão não tem valor e nada será impresso. Esta é uma forma extremamente económica de fazer a escolha de assuntos a serem impressos. (Para mais informações sobre o estranho comportamento das expressões lógicas, ver cap. 13 do manual do *Spectrum*.)

Linhas 2700-2710. Estas duas variáveis serão utilizadas para registar os totais dos vários grupos de subdenominações e do total final.

Linhas 2720-2910. Neste desvio, cada acesso de A$ é examinado à vez, para ver se se trata de um só assunto, de uma denominação principal ou de uma subdenominação. Se for um simples assunto, a denominação será impressa e a quantia introduzida na coluna principal. Se for uma denominação principal, não será impressa qualquer quantia correspondente. Se for uma subdenominação, a denominação será identificada por dois espaços e a quantia a

ela associada impressa numa coluna separada, em frente da coluna principal. No final de cada grupo de subdenominações, o total do grupo será impresso ao fundo deste.

Ensaio do módulo 2.2.8

Introduza algumas denominações e imprima-as no *écran*. Todos os pontos decimais da coluna principal devem estar em linha.

MÓDULO 2.2.9

```
2120 REM ************************
2130 REM CHANGES AND DELETIONS
2140 REM ************************
2150 FOR I=1 TO C(CD)-1
2160 PRINT AT 0,10;"CHANGE OR DE
LETE"
2170 IF A$(CD,I,1)<>"£" THEN PRI
NT AT 3,0;O$
2180 PRINT AT 5,0;O$
2190 IF A$(CD,I,1)<>"£" THEN PRI
NT AT 3,0;A$(CD,I,2 TO )
2200 IF A$(CD,I,1)="£" THEN PRIN
T AT 5,0;A$(CD,I,2 TO )
2210 IF A(CD,I)=0 THEN GO TO 225
0
2220 LET M=A(CD,I)
2230 GO SUB 2980
2240 PRINT TAB 16;M$
2250 PRINT AT 18,0;">ENTER=NEXT
ITEM"
2260 PRINT ">""CCC""=CHANGE AMOU
NT"
2270 PRINT ">""ZZZ""=QUIT FUNCTI
ON"
```

```
2280 PRINT ">""DDD""=DELETE ITEM
"
2290 INPUT Q$
2300 IF Q$="ZZZ" THEN RETURN
2310 IF Q$="DDD" THEN GO SUB 246
0: RETURN
2325 IF Q$="" THEN GO TO 2440
2330 PRINT AT 18,0;O$;O$;O$;O$
2340 PRINT AT 18,0;"AMOUNT TO BE
 ADDED?";
2350 INPUT Q$
2360 PRINT Q$
2370 INPUT "Is that correct? (Y/
N)";R$
2390 PRINT AT 18,0;O$;O$;O$;O$
2400 IF R$="N" THEN GO TO 2340
2410 LET A(CD,I)=A(CD,I)+VAL Q$
2420 PRINT AT 6,0;O$
2430 GO TO 2160
2440 NEXT I
2450 RETURN
```

Este módulo permite ao utilizador modificar as quantias associadas a assuntos já introduzidos nas contas. Não há qualquer função de busca ligada ao módulo; o utilizador apenas passará em revista os assuntos das contas, até chegar ao assunto correcto. A quantia associada a um assunto é modificada através da inserção de uma quantia que deverá ser adicionada à quantia corrente armazenada. Parte-se do princípio de que as alterações terão como objectivo, regra geral, o registo de despesas ou rendimentos extra, sob denominações já existentes. Se desejar subtrair, bastará inserir um número negativo.

Ensaio do módulo 2.2.9

Tente alterar alguns assuntos, mostrando depois os respectivos valores.

MÓDULO 2.2.10

```
2460 REM ************************
2470 REM DELETE ITEM
2480 REM ************************
2490 LET PLACE=I
2500 LET DELETION=1
2510 IF A$(CD,PLACE,1)<>"*" THEN
 GO TO 2550
2520 LET DELETION=0
2530 LET DELETION=DELETION+1
2540 IF A$(CD,PLACE+DELETION,1)=
"£" THEN GO TO 2530
2550 FOR K=PLACE TO C(CD)-DELETI
ON-1
2560 LET A$(CD,K)=A$(CD,K+DELETI
ON)
2565 LET A(CD,K)=A(CD,K+DELETION
)
2570 NEXT K
2580 LET C(CD)=C(CD)-DELETION
2650 RETURN
```

Este módulo apaga assuntos das contas. Se o assunto a ser apagado for uma denominação principal, todas as subdenominações que lhe estiverem associadas serão igualmente apagadas.

Comentário

Linhas 2490-2540. Se o assunto a ser apagado for uma denominação principal, a variável DELETION (apagamento) será aumentada até ser igual à denominação principal e ao número das subdenominações.

Linhas 2550-2580. Os assuntos colocados mais acima na disposição são movidos para baixo, para cobrir os assuntos a ser apagados. A variável que regista o número de assuntos é decrescida pelo número de assuntos apagados.

Ensaio do módulo 2.2.10

Apague alguns assuntos e verifique a eficiência dessa acção. O programa está agora pronto para ser utilizado.

Avançando mais

1) Por que não tentar acrescentar mais algumas cores interessantes?

2) Tente acrescentar uma função que calcule o balanço e inclua um ou outro lado das contas.

3) Se houvesse, na realidade, cinquenta ou mais assuntos num dos lados das contas, levaria muito tempo percorrê-las para fazer alterações. Experimente acrescentar uma função de busca que permita ao utilizador inserir um número, saltando depois esse número de assuntos, para a frente ou para trás.

2.3 BANQUEIRO

Neste programa, começamos por fazer extenso uso de linhas multienunciado. O programa é um instrumento claro, que permite manter os seus próprios registos financeiros, de forma muito aproximada à de um extracto bancário. Trata de pagamentos periódicos, quer regulares quer irregulares, inserindo-os no enunciado do dia certo de cada mês em que o pagamento tem lugar.

Este programa é simples mas, comparado com o que já se fez antes, não é tão curto como parece. Seria 50% mais extenso se não fosse o facto de serem utilizadas linhas multienunciado.

Uma das coisas a ter em atenção, ao introduzir o programa, é a utilização de enunciados IF. Tais enunciados são capazes de causar destruição, se usados impropriamente em linhas multienunciado, criando dificuldades no programa que são extremamente difíceis de localizar. Igualmente podem ser utilizadas linhas multienunciado para aumentar a eficência e facilidade com que são usados os enunciados IF. A causa disto é que, se a condição especificada por um enunciado IF não for satisfeita, o programa não passará por cima dessa parte da linha que contém o enunciado IF, mas sim por cima de todo o resto da linha.

Por outras palavras, quaisquer enunciados posteriores ao enunciado IF apenas serão executados se o enunciado IF for verdadeiro. Isto é tão diferente do comportamento das linhas de enunciado único que é fácil de esquecer.

A vantagem disto é que possibilita uma forma automática e elegante de GOTO, que nem mesmo precisa de ser especificada. Se tiver uma série de dez operações que devem ser levadas a cabo em conjunto, desde que, digamos, C = 1 em algum ponto, então, com linhas de enunciado único, teria de colocar um enunciado IF no início da secção, para especificar o rodeio da secção, caso C não seja igual a 1. Isto funciona, mas parece trapalhão e torna-se difícil ler um programa onde há muitos rodeios.

Com linhas multienunciado, no entanto, pode começar uma linha única com IF C = 1 e segui-la de dez operações. Isto não só funcionará e poupará memória, como também tornará o programa mais legível em muitas formas, posto que será imediatamente claro que as dez operações formam uma unidade lógica.

```
1000 REM ************************
1010 REM MENU
1020 REM ************************
1030 CLS : INK 0: PAPER 7: PRINT
 AT 0,10: INK 2; FLASH 1;"BANKER
"
1040 PRINT '"1)NEW PAYMENTS"
1050 PRINT '"2)EXAMINE/DELETE PA
YMENTS"
1060 PRINT '"3)PRINT STATEMENT"
1070 PRINT '"4)INITIALISE"
1080 PRINT '"5)STOP"
1090 INPUT Z$: CLS
1100 IF Z$="1" THEN GO SUB 1250
1110 IF Z$="2" THEN GO SUB 1420
1120 IF Z$="3" THEN GO SUB 1550
1130 IF Z$="4" THEN GO SUB 1180
1140 IF Z$="5" THEN GO TO 1160
1150 CLS : GO TO 1000
1160 PRINT AT 10,10; FLASH 1; IN
K 2;"BANKER": INPUT "HAVE YOU EN
TERED ANY NEW DATA   YOU WANT TO
 SAVE? (Y/N)";Q$: IF Q$="Y" THEN
 SAVE "BANKER": BEEP 1,2: PRINT
"REWIND THEN ""ENTER"" TO VERIFY
": PAUSE 0: VERIFY "BANKER": PRI
NT "        PROGRAM VERIFIED"
1170 STOP
```

Trata-se de um módulo de ementa padrão.


```
1180 REM ************************
1190 REM VARIABLES
1200 REM ************************
1210 DIM A$(100,26): DIM A(100)
1220 DEF FN A$(X)=(STR$ (X+10000
.001))(2 TO 8)
1230 LET PAYMENTS=0
1240 RETURN
```

Comentário

Linha 1210. A$ será usada para armazenar os nomes e outras informações acerca dos pagamentos individuais. As quantias reais serão armazenadas por A.

Linha 1220. Se seguiu os dois últimos programas, não deve ter dificuldade em identificar o objectivo desta função. Numa linha única, esta função cria um formato padrão para qualquer número até 9999,99 que fique disponível. Deve lembrar-se de que, em relação a «Unificha», módulo 2, discutimos os tipos de utilização onde uma função definida pelo utilizador resultava numa significativa economia, comparada com uma sub-rotina de uma linha ou duas. Esta função, com o seu único argumento, X, será usada para formar duas diferentes variáveis, economizando assim o uso de duas sub-rotinas curtas. Quando se tiver familiarizado com o uso delas neste programa, poderá querer regressar às duas anteriores e substituir as duas sub-rotinas curtas utilizadas para padronizar o formato dos números.

Linha 1230. PAYMENTS (pagamentos) é utilizada para registar o número total de assuntos da ficha.

```
1250 REM ************************
1260 REM NEW REGULAR PAYMENT
1270 REM ************************
1280 PRINT AT 0,10; FLASH 1; INK
 2;"PAYMENTS"
1290 PRINT '"1) CREDIT / 2)DEBIT
? ";: INPUT CD: PRINT CD
1300 PRINT '"NAME OF PAYMENT? ";
: INPUT Q$: PRINT Q$
1310 PRINT '"AMOUNT? ";: INPUT Q
: PRINT Q
1320 PRINT '"MONTHS:e.g. 0104071
0 : ";: INPUT R$: PRINT R$
1330 PRINT '"DAY OF PAYMENT? ";:
 INPUT S: PRINT S
1340 INPUT "ARE THESE CORRECT? (
Y/N)";T$: IF T$="N" THEN CLS : G
O TO 1250
1350 LET PAYMENTS=PAYMENTS+1: FO
R J=PAYMENTS TO 2 STEP -1
1360 IF S<CODE A$(J-1,1) THEN LE
T A$(J)=A$(J-1): LET A(J)=A(J-1)
: NEXT J
1370 LET A$(J)="": LET A$(J,26)=
CHR$ CD: LET A$(J,14 TO 25)=Q$:
LET A(J)=Q
1380 IF CD=2 THEN LET A(J)=A(J)*
-1
1390: FOR I=1 TO LEN R$ STEP 2:
LET A$(J,1+VAL R$(I TO I+1))="1"
: NEXT I:
1400 LET A$(J,1)=CHR$ S
1410 RETURN
```


Este módulo aceita a inserção de novos assuntos, incluindo pormenores de nome, quantia, mês em que é feito o pagamento e dia do pagamento. O módulo também anota se o assunto é um crédito ou um débito. Se for um débito, a quantia será armazenada de forma negativa.

Comentário

Linhas 1350-1360. Começando pelo último assunto da ficha, o módulo passa em revista os assuntos para encontrar o primeiro assunto alfabeticamente abaixo do assunto a ser introduzido. Isto selecciona os assuntos por ordem de dia de pagamento, visto que o dia de pagamento é registado no código de caracteres do primeiro carácter de cada linha, em **A$**. Note-se que, ao inserir assuntos simples numa disposição, é sempre mais eficiente começar pelo fim da lista de assuntos, já que cada assunto pode então ser examinado e, se necessário, deslocado um lugar de cada vez até ser encontrada a posição correcta. Isto elimina a necessidade de dois desvios separados, um para procurar na ficha a posição correcta e outro para deslocar os elementos, fazendo espaço para o novo acesso.

Linhas 1370-1400. A quantia é armazenada em **A**. Toda a restante informação é armazenada na mesma linha, em **A$**. O primeiro carácter é utilizado para armazenar o dia do pagamento: os doze seguintes são usados para registar os meses em que é feito o pagamento. As posições 14 a 25 são utilizadas para o nome e o assunto. A posição 26 regista se o assunto é um crédito ou um débito.

Ensaio do módulo 2.3.3

Passe o programa e solicite a função das iniciais. Introduza agora alguns assuntos e pare. Imprima no *écran* os conteúdos de **A$** e **A** e tente compará-los com a descrição dada no comentário deste módulo.

```
1550 REM ************************
1560 REM COMPILE STATEMENT
1570 REM ************************
1580 LET TOTAL=0
1590 PRINT AT 0,10; INK 2; FLASH
 1;"STATEMENT"
1600 PRINT '''"NUMBER OF MONTH FO
R STATEMENT:": INPUT Q: PRINT Q
1610 CLS : PRINT AT 0,10; INK 1;
"MONTH: ";Q
1620 FOR I=1 TO PAYMENTS
1630 IF A$(I,1+Q)<>"1" THEN GO T
O 1690
1640 IF A$(I,26)=CHR$ 1 THEN PAP
ER 6
1650 PRINT 'CODE A$(I,1);"*" AND
 A$(I,26)=CHR$ 1;TAB 3;":";A$(I,
14 TO 25);
1660 PRINT TAB 16;FN A$(ABS A(I)
);"_";: LET TOTAL=TOTAL+A(I): IF
 TOTAL<0 THEN INVERSE 1
1670 PRINT FN A$(ABS TOTAL): INV
ERSE 0
1680 PAPER 7
1690 NEXT I: INPUT """ENTER"" TO
 CONTINUE";Q$
1700 RETURN
```

Este módulo aceita uma entrada especificando um mês, mostrando depois as contas de todos os pagamentos e receitas durante esse mês.

Comentário

Linha 1630. É ignorado qualquer assunto que não tenha um 1 na posição do carácter correspondente ao mês especificado.

Linha 1640. A cor PAPER é passada a amarelo, para os assuntos de crédito.

Linha 1650. Note-se o uso de AND para imprimir um asterisco a seguir à data dos assuntos de crédito.

Linhas 1660-1670. FN A$ é aplicada a ABS A(I) e ABS TOTAL. Note-se o uso de ABS aqui — a função daria um resultado incompreensível se aplicada a um número negativo, devido à presença do sinal menos. A ABS poderia, é claro, ser incluída na função.

Ensaio do módulo 2.3.4

Imprima no *écran* as contas dos vários meses.

MÓDULO 2.3.5

```
1420 REM ************************
1430 REM DELETE PAYMENTS
1440 REM ************************
1450 FOR I=1 TO PAYMENTS
1460 CLS : IF CODE A$(I,26)=1 TH
EN PRINT ' PAPER 6;"CREDIT"
1470 IF CODE A$(I,26)=2 THEN PRI
NT ' PAPER 6;"DEBIT"
1480 PRINT "PAYMENT:";A$(I,14 TO
 25)''"AMOUNT:";FN A$(ABS A(I))
1490>PRINT '"MONTHS:";: FOR J=1
TO 12: PRINT STR$ J AND A$(I,1+J
)="1";"/";: NEXT J
1500 PRINT ''"DAY OF PAYMENT:";C
ODE A$(I,1)
1510 PRINT PAPER 6;'"COMMANDS:"'
"1) ""DDD"" DELETE"'"2) ""ZZZ""
QUIT"'"3) ""ENTER"" FOR NEXT ITE
M."
1520 INPUT Q$: IF Q$="DDD" THEN
```

```
FOR J=I TO PAYMENTS: LET A$(J)=A
$(J+1): LET A(J)=A(J+1): NEXT J:
 LET PAYMENTS=PAYMENTS-1: RETURN
1530 IF Q$="" THEN NEXT I: RETUR
N
1540 RETURN
```

Este módulo mostra assuntos da ficha e dá ao utilizador a opção de os apagar.

Comentário

Linha 1520. Os assuntos são deslocados para baixo, na ficha, para cobrir aquele que vai ser apagado. Note-se que isto significa que o assunto final da disposição está agora duplicado, visto que nada foi feito para o cobrir. Isto é irrelevante, uma vez que PAYMENTS foi decrescida de 1 e a posição do que foi o último assunto da ficha se torna invisível para o programa.

Ensaio do módulo 2.3.5

Se a função resultar, o programa está agora completo.

Sumário

Trata-se realmente de um programa muito simples, que levanta a interessante questão de até onde deveríamos ir no assunto da complexidade de um programa para simplificar a sua utilização. Programas sofisticados de contas bancárias oferecem muitas vezes vantagens no registo das mudanças de ano, frequência de pagamentos, datas finais, etc. Neste programa, adoptámos o simples expediente de perguntar ao utilizador em que mês vai ser feito o pagamento, eliminando assim a necessidade de distinguir entre pagamentos em prestações e uma variedade de ordens de importância diversa. Parece-me que, a menos que as suas transacções financeiras sejam extremamente complexas, não quererá dar-se ao trabalho

de escrever novas funções programáticas, que tratem dos pagamentos mensais sem que o utilizador tenha de especificar os meses. Tenha cuidado com a complexidade em excesso nos seus próprios programas; se devotar dez ou vinte linhas extra a funções que o cérebro humano poderia efectuar mais depressa do que a introdução dos dados, estará a perder tempo e esforços.

Avançando mais

1) O programa não tem qualquer dispositivo para levar por diante um balanço de mês para mês. Decida-se a acrescentar-lhe um.

2) O asterisco que acompanha a data dos assuntos de crédito está lá para destacar o assunto ao ser enviado para a impressão, a qual não registará a diferente cor do papel. Duplique os enunciados de impressão, no módulo 4, com LPRINT.

3) Será capaz de combinar a instrução de inverter a impressão de um número negativo na função FN A$?

CAPÍTULO 3

DESENHO DE UMA IMAGEM — GRÁFICOS NO «SPECTRUM»

Neste capitulo examinaremos algumas das capacidades gráficas do *Spectrum*. Para esgotar este assunto seria preciso um livro, e não é feita aqui qualquer tentativa nesse sentido. Concentrar-nos-emos em gráficos práticos, que possam contribuir para a eficácia de uma variedade de programas. Deixaremos de lado áreas tão fascinantes como a dos gráficos tridimensionais, gráficos móveis e modelos criados por funções matemáticas. Não quero dizer com isto que tais áreas não sejam importantes; apenas que seria impossivel render-lhes aqui inteira justiça.

Nos programas seguintes examinaremos as possibilidades que se abrem através dos gráficos definidos pelo utilizador e suas capacidades, tentando ultrapassar alguns dos problemas da criação e armazenamento de desenhos a todo o *écran*.

Os programas apresentados neste capitulo são:

1) *Caracteres* — um programa concebido para lhe permitir a criação dos seus próprios caracteres definíveis pelo utilizador, com aplicação em outros programas.

2) *Dicionário* — um método para armazenar os muitos caracteres dos gráficos definidos pelo utilizador que vai criar.

3) *Quebra-cabeças* — um programa que organiza o desenho de formas utilizando o comando DRAW (desenho).

4) *Artista* — um programa que lhe permite desenhar figuras a todo o *écran*, utilizando todos os caracteres gráficos do *Spectrum*, incluindo os definidos pelo utilizador, armazenando as figuras resultantes, juntamente com as suas características de cor.

5) *Desenhista* — um programa que permite formar um desenho até ao limite de 65536*65536 *pixels*[1], acrescentar e apagar, examinar o desenho a várias escalas e rodá-lo, no todo ou em parte, dentro dos limites do *écran*.

3.1 CARACTERES

Alguns de vós podem já ter escrito um programa definidor de caracteres. Este é, no entanto, um utensílio claro para aqueles que ainda o não fizeram e também uma apresentação às técnicas que usaremos em outros programas.

O programa possibilita-lhe desenhar, em larga escala, grupos de até quatro caracteres gráficos, apagar caracteres gráficos já existentes e salvar, em fita, o carácter assim criado.

Se não está familiarizado com a ideia de gráficos definidos pelo utilizador, deve voltar a ler o cap. 14 do manual do *Spectrum*, antes de tentar introduzir este programa.

MÓDULO 3.1.1

```
1000 REM ************************
1010 REM MENU
1020 REM ************************
1030 PAPER 7: INK 0: CLS
1040 GO SUB 2400
1050 PRINT "          CHARACTERS
"
1060 PRINT '"FUNCTIONS AVAILABLE
:"
```

[1] *Pixel:* termo, criado a partir da contracção das palavras inglesas *picture elements*, que designa cada um dos minúsculos pontos luminosos que compõem qualquer imagem num *écran* do tipo televisivo. *(N. do T.)*


```
1070 PRINT '"1)INITIALISE"
1080 PRINT '"2)CREATE NEW CHARAC
TERS"
1090 PRINT '"3)SAVE CHARACTERS"
1100 PRINT '"4)DELETE CHARACTERS
"
1110 PRINT '"5)STOP"
1120 PRINT '"WHICH DO YOU REQUIR
E?"
1130 INPUT Z$
1140 CLS
1150 IF Z$="1" THEN GO SUB 1230
1160 IF Z$="2" THEN GO SUB 1290
1170 IF Z$="3" THEN GO SUB 2270
1180 IF Z$="4" THEN GO SUB 2040
1190 IF Z$="5" THEN STOP
1200 PAPER 7: CLS
1210 GO TO 1000
1220 STOP
```

Trata-se de um módulo de ementa padrão.

MÓDULO 3.1.2

```
2400 REM ************************
2410 REM LOAD INITIAL CHARACTER
2420 REM ************************
2430 RESTORE
2440 FOR I=0 TO 7
2450 READ BYTE
2460 POKE USR CHR$ 144+I,BYTE
2470 NEXT I
2480 RETURN
2490 DATA 255,129,129,129,129,12
9,129,255
```

Este módulo passa à memória o carácter «□», na posição ocupada por «A» na área dos gráficos definidos pelo utilizador. O carácter será utilizado no decurso do programa.

Comentário

Linhas 2440-2470. Se já voltou a ler o cap. 14 do manual do *Spectrum*, não deve ter dificuldade em compreender o que aqui se passa. Os números no enunciado de dados representam valores binários que, quando armazenados na área de caracteres definidos pelo utilizador, formarão o carácter «□». 255 passa a sistema binário como 11111111 e 129 como 10000001. São lidos individualmente e colocados na memória. CHR$ é «A» na área definida pelo utilizador.

Ensaio do módulo 3.1.2

Depois de introduzir e passar este módulo, «A» no modo gráfico deverá imprimir o carácter «□».

MÓDULO 3.1.3

```
1230 REM ************************
1240 REM VARIABLES
1250 REM ************************
1260 DIM Z(26,8)
1270 LET CHARACTERS=1
1280 RETURN
```

Estão aqui todas as variáveis.

Comentário

Linha 1260. Os números que serão utilizados para criar os caracteres definidos pelo utilizador são armazenados nesta disposição, para facilidade de manipulação. Quaisquer apagamentos são realizados em primeiro lugar na disposição, a qual é depois levada até à memória.

MÓDULO 3.1.4

```
1290 REM ************************
1300 REM DRAW GRID
1310 REM ************************
1330 FOR I=2 TO 17
1340 FOR J=6 TO 21
1350 PRINT AT I,J;"A"
1360 NEXT J
1370 NEXT I
1380 FOR I=6 TO 21
1390 PRINT OVER 1; PAPER 5;AT 2,
I;" ";AT 9,I;" ";AT 10,I;" ";AT
17,I;" "
1400 NEXT I
1410 FOR I=2 TO 17
1420 PRINT OVER 1; PAPER 5;AT I,
6;" ";AT I,13;" ";AT I,14;" ";AT
 I,21;" "
1430 NEXT I
1440 PRINT AT 20,0;"5,6,7,8 TO M
OVEA1 TO INK INA0 TORUB OUTA9 TO
 REGISTER CHARACTERS"
```

Este módulo desenha uma grelha de 16*16 dentro da qual os caracteres definidos pelo utilizador serão construídos. A grelha é elaborada a partir do carácter definido no módulo 2.

Comentário

Linhas 1380-1430. Existem quatro quadrados de caracteres separados. Para ajudar a distingui-los, as respectivas margens encontram-se sombreadas a azul, utilizando a característica de sobreposição para evitar desgastar a grelha.

Ensaio do módulo 3.1.4

O módulo deverá imprimir uma grelha de 16*16, sombreada no interior pelas margens de quatro quadrados de 8*8.

MÓDULO 3.1.5

```
1450 REM **********************
1460 REM FILL IN SQUARES
1470 REM **********************
1480 DIM A$(4,8,8)
1490 LET X=2: LET Y=6
1500 PRINT AT X,Y; PAPER 8; OVER
 1;"*": PAUSE 5
1510 PRINT AT X,Y; PAPER 8; OVER
 1;"*"
1520 LET T$=INKEY$
1530 IF T$="" THEN GO TO 1500
1540 BEEP .05,40
1550 LET X1=X-1-8*(X>9): LET Y1=
Y-5-8*(Y>13)
1560 LET XY=1+(Y>13)+2*(X>9)
1570 IF T$="0" THEN LET A$(XY,X1
,Y1)="A"
1580 IF T$="1" THEN LET A$(XY,X1
,Y1)="_"
1590 IF T$="9" THEN GO TO 1660
1600 PRINT PAPER 8;AT X,Y;"A";:
IF A$(XY,X1,Y1)="_" THEN PRINT P
APER 8;AT X,Y;"_"
1610 LET X=X+(T$="6")-(T$="7")
1620 LET X=X+(X<2)-(X>17)
1630 LET Y=Y-(T$="5")+(T$="8")
1640 LET Y=Y+(Y<6)-(Y>21)
1650 GO TO 1500
```


Este módulo é o fulcro do programa. O seu objectivo é permitir ao utilizador mover um cursor intermitente pela grelha, quer tingindo quadrados quer apagando quadrados já tintos. Quando o carácter estiver definido a seu contento, poderá largar o módulo e examinar esse carácter na sua correcta dimensão.

Comentário

Linha 1480. É muito mais fácil manipular o conteúdo de uma disposição do que o que se apresenta no écran. Portanto, quando um quadrado da grelha vai ser tingido, é registado, em primeiro lugar, na disposição **A$**, e é esta disposição que será impressa no *écran*. Note-se que a disposição **A$** leva em conta, nas suas dimensões, a existência de quatro quadrados separados de caracteres.

Linha 1490. São estas as coordenadas do canto superior esquerdo da grelha.

Linhas 1500-1530. Esta rotina imprime um cursor intermitente na posição X,Y até ser premida uma tecla. PAPER é colocado em 8, para que a tonalidade do quadrado fique inalterável. Visto que o asterisco é impresso OVER (sobre) duas vezes, não tem qualquer efeito no conteúdo do quadrado em grelha.

Linha 1540. A função INKEY$ não activa o sinal sonoro do teclado, pelo que este deve ser substituído, para assinalar o registo de uma tecla.

Linhas 1550-1560. Esta rotina é extremamente útil quando se torna necessário movimentar um cursor pelo *écran* sob controlo do utilizador. O uso das condições lógicas entre parênteses altera o valor da coordenada X ou Y, se uma das teclas de seta do cursor for premida. As linhas 1620 e 1640 verificam se o cursor não saiu dos limites desejados... o que, noutras circunstâncias, poderiam ser os limites do *écran*. Se os limites foram excedidos de algum modo, o cursor é enviado de volta, usando novamente uma condição lógica como variável. Estas duas verificações serão encontradas vezes sem conta em programas que fazem deslocar algo no *écran*.

Ensaio do módulo 3.1.5

Passe o programa, coloque-o no ponto de partida, solicite a função 2 e desloque o cursor intermitente pela grelha, colorindo e apagando à vontade.

MÓDULO 3.1.6

```
1660 REM ************************
1670 REM RECORD CHARACTERS
1680 REM ************************
1690 PAPER 7
1700 INPUT "Is the Pattern satis
factory?    (ZZZ) TO QUIT";Q$
1710 IF Q$="ZZZ" THEN RETURN
1720 IF Q$="Y" THEN GO TO 1740
1730 PAPER 5: GO TO 1490
1740 INPUT "WHICH CELLS TO BE SA
VED? (ONE   DIGIT PER CELL, ANY
ORDER):";Q$
1750 LET CELLS=LEN Q$
1760 IF CHARACTERS+CELLS<=21 THE
N GO TO 1790
1770 PRINT AT 20,0;"NOT ENOUGH R
OOM."
1780 RETURN
1790 FOR I=1 TO CELLS
1800 FOR J=1 TO 8
1810 LET BYTE=0
1820 FOR H=1 TO 8
1830 IF A$(VAL Q$(I),J,H)="_" TH
EN LET BYTE=BYTE+2↑(8-H)
1840 NEXT H
1850 LET Z(CHARACTERS+I-1,J)=BYT
E
1860 NEXT J
1870 NEXT I
1890 LET CHARACTERS=CHARACTERS+C
ELLS
```


Este módulo transfere o modelo criado na grelha para a disposição Z, sob a forma de uma série de números que, quando colocados na área da memória de caracteres definidos pelo utilizador, reproduzirão o carácter que acabou de desenhar-se.

Comentário

Linhas 1790-1870. Com efeito, cada linha de **A$** é tratada como um número binário de oito digitos, com cada quadrado colorido representando um «1». O número resultante é colocado no espaço relevante da disposição Z.

Ensaio do módulo 3.1.6

Pode introduzir um carácter e verificar se valores sensíveis foram introduzidos na disposição Z.

MÓDULO 3.1.7

```
1900 REM ************************
1910 REM ACTIVATE CHARACTERS
1920 REM ************************
1930 FOR I=1 TO 20
1940 FOR J=0 TO 7
1950 POKE USR CHR$ (144+I)+J,Z(I
,J+1)
1960 NEXT J
1970 NEXT I
1980 INPUT "Do you want to test
characters?";Q$
1990 IF Q$<>"Y" THEN RETURN
2000 PRINT "PRESS ANY KEY TO STA
RT, THEN    CHANGE TO GRAPHICS M
ODE."
2010 PAUSE 0
2020 CLS : INPUT Q$
2030 RETURN
```

Os números armazenados na disposição Z são colocados na memória dos caracteres definidos pelo utilizador.

Comentário

Linhas 1930-1970. Esta é uma função similar à levada a cabo pelo módulo 2.

Linha 2020. A inserção da sequência nesta linha é apenas uma oportunidade para introduzir alguns caracteres e verificar os nossos caracteres definidos pelo utilizador, previamente armazenados. A sequência não tem qualquer outro objectivo.

Ensaio do módulo 3.1.7

Está agora em posição de poder introduzir vinte dos seus próprios caracteres, verificando se eles foram correctamente armazenados na memória.

MÓDULO 3.1.8

```
2040 REM ************************
2050 REM DELETE CHARACTERS
2060 REM ************************
2070 IF CHARACTERS=1 THEN RETURN

2080 FOR I=2 TO CHARACTERS
2090 PRINT CHR$ (143+I)
2100 PRINT ""DO YOU WISH TO DEL
ETE THIS       CHARACTER? (Y/N)"
2110 INPUT Q$
2120 IF Q$="Y" THEN GO TO 2160
2130 CLS
2140 NEXT I
2150 RETURN
```



```
2160 FOR J=I-1 TO CHARACTERS-2
2170 FOR K=1 TO 8
2180 LET Z(J,K)=Z(J+1,K)
2190 NEXT K
2200 NEXT J
2210 FOR I=1 TO 8
2220 LET Z(J,I)=0
2230 NEXT I
2240 LET CHARACTERS=CHARACTERS-1
2250 GO SUB 1900
2260 RETURN
```

Este módulo dá ao utilizador a possibilidade de apagar caracteres que tenham sido armazenados anteriormente. O apagamento é conseguido pela remoção dos elementos relevantes da disposição Z, voltando a inserir os caracteres na memória.

Ensaio do módulo 3.1.8

Experimente apagar alguns caracteres, examinando depois os restantes, para se certificar de que eles não foram de alguma forma corrompidos pelo processo.

MÓDULO 3.1.9

```
2270 REM ************************
2280 REM SAVE CHARACTERS
2290 REM ************************
2295 LET Y$="": FOR I=1 TO CHARA
CTERS-1: FOR J=1 TO 8: LET Y$=Y$
+CHR$ Z(I,J): NEXT J: NEXT I
2300 DIM X$(LEN Y$): LET X$=Y$
2305 FOR I=1 TO CHARACTERS-1
2310 PRINT CHR$ (144+I);" ";
```

```
2320 NEXT I
2330 PRINT '"WHAT NAME DO WANT T
O GIVE TO    THIS SET OF CHARACT
ERS?"
2340 INPUT N$: PRINT '''N$
2350 SAVE N$CODE USR "A",21*8
2355 SAVE N$ DATA X$()
2360 CLS : PRINT '''"NOW PLEASE R
EWIND TO BEGINNING, START TAPE,
THEN PRESS ANY KEY."
2370 PAUSE 0
2380 VERIFY N$CODE USR "A",21*8
2385 VERIFY N$ DATA X$()
2390 PRINT '''"CHARACTER SET VERI
FIED.": PAUSE 200: RETURN
```

O conjunto de caracteres criado pelo utilizador é baptizado e salvo para uma cassete sob duas formas, uma das quais se relaciona com o próximo programa deste capítulo.

Comentário

Linha 2295. Os valores de Z são transferidos, sob a forma de códigos de carácter único, para a sequência **Y$**. Uma vez que o *Spectrum* não salvará uma sequência de dimensões tão indefinidas, **X$** é dimensionada para a mesma medida que **Y$** e o conteúdo de **Y$** transferido.

Linhas 2300-2340. Esta secção imprime o conjunto de caracteres, na sua totalidade, e convida o utilizador a dar-lhe um nome.

Linha 2350. Esta linha instrui o *Spectrum* para salvar o conteúdo da sua memória, começando pelo início da área de caracteres definidos pelo utilizador e incluindo 168 *bytes* de memória, ou seja, toda a memória dedicada aos caracteres definidos pelo utilizador.

Linha 2355. O conjunto de caracteres é igualmente salvo, sob a forma de **X$**.

Linha 2380. O conjunto de caracteres armazenado em fita pode ser verificado do mesmo modo que um programa, e é sensato fazê-lo, para o caso de ter havido alguma falha na gravação dos seus caracteres arduamente conseguidos.

Ensaio do módulo 3.1.9

Crie um conjunto de caracteres e salve-o, utilizando este módulo. Certifique-se de que salvou o programa, cortando depois a corrente, momentaneamente, para limpar o conteúdo da memória. Introduza agora LOAD N$ — utilizando o mesmo nome que em N$ — CODE USR «A», 21*8 e reproduza aquilo que salvou. Quando a introdução estiver completa, deverá descobrir que os novos caracteres voltaram. Se assim for, o programa foi correctamente introduzido e está pronto a ser utilizado.

Sumário

Para lá da utilidade na criação de caracteres a utilizar em outros programas, pôde encontrar neste algumas técnicas que vai retomar repetidas vezes em programas de gráficos. Nelas se inclui o cursor intermitente, a utilização das teclas do cursor para fazer movimentar um carácter no *écran*, o uso de condições lógicas para delimitar esse movimento e a utilização de disposições que simulam o *écran*.

Avançando mais

1) Decida-se a aumentar o número de caracteres que podem ser definidos num bloco de quatro a seis.

2) Faça o programa imprimir a dimensão correcta do carácter em desenvolvimento, juntamente com a grelha, à medida que vão sendo construídos.

3.2 DICIONÁRIO

Este curto programa aumenta grandemente a utilidade do gerador de caracteres, permitindo ao utilizador criar um dicionário de todos os caracteres definidos pelo utilizador previamente criados. O utilizador pode então valer-se do conjunto desta reserva para criar novas combinações de caracteres em conjuntos de caracteres.

MÓDULO 3.2.1

```
1000 REM ************************
1010 REM MENU
1020 REM ************************
1030 INK 0: PAPER 7: CLS : PRINT
 "        CHARACTER DICTIONARY"
1040 PRINT '"1) INITIALISE"
1050 PRINT '"2) LOAD NEW CHARACT
ERS"
1060 PRINT '"3) CREATE NEW SET"
1070 PRINT '"4) SAVE SET OF CHAR
ACTERS"
1080 PRINT '"5) STOP"
1090 INPUT Z$: CLS
1100 IF Z$="1" THEN GO SUB 1180
1110 IF Z$="2" THEN GO SUB 1430
1120 IF Z$="3" THEN GO SUB 1220
1130 IF Z$="4" THEN GO SUB 1360
1140 IF Z$="5" THEN GO TO 1160
1150 GO TO 1030
1160 INPUT "DO YOU WISH TO RE-RE
CORD? (Y/N)";Q$: IF Q$="Y" THEN
SAVE "DICTIONARY": BEEP 1,40: PR
INT AT 10,0;"REWIND, THEN ANY KE
Y TO VERIFY.": PAUSE 0: VERIFY "
DICTIONARY": PRINT '"PROGRAM VER
IFIED"
1170 STOP
```


Um módulo de ementa padrão.

MÓDULO 3.2.2

```
1180 REM ************************
1190 REM INITIALISE
1200 REM ************************
1210 LET P$="": RETURN
```

O módulo de partida.

Comentário

Linha 1210. Esta sequência é usada para armazenar os valores que serão empurrados para a área da memória definida pelo utilizador.

MÓDULO 3.2.3

```
1430 REM ************************
1440 REM LOAD NEW CHARACTERS
1450 REM ************************
1460 INPUT "NAME OF CHARACTER SE
T? ";N$
1470 LOAD N$CODE USR "A",168: LO
AD N$ DATA T$( )
1480 FOR I=0 TO 20: PRINT CHR$ (
144+I);" ";: NEXT I
1490 INPUT "DO YOU WANT THIS SET
? (Y/N)";Q$: IF Q$="N" THEN RETU
RN
1500 LET P$=P$+T$: RETURN
```

Este módulo transfere da fita magnética os conjuntos de caracteres criados pelo anterior programa e acrescenta-os ao dicionário.

Ensaio do módulo 3.2.3

Com este módulo no lugar, deverá poder transferir conjuntos de caracteres da fita. P$ deverá ser oito vezes o número de caracteres transferidos.

MÓDULO 3.2.4

```
1220 REM ************************
1230 REM PRINTCHARACTERS
1240 REM ************************
1250 FOR I=0 TO 167: POKE USR "A
"+I,0: NEXT I
1260 LET C$=""
1270 FOR I=1 TO LEN P$/160+1: PR
INT AT 0,10;"PAGE ";I: FOR J=1 T
O 20: FOR K=1 TO 8
1280 IF 160*(I-1)+8*(J-1)+K>LEN
P$ THEN GO TO 1310
1290 POKE USR "A"+(K-1),CODE P$(
160*(I-1)+8*(J-1)+K): NEXT K
1300 PRINT 20*(I-1)+J;" ";CHR$ 1
44: NEXT J
1310 INPUT "CHAR.NO, N=NEXT PAGE
_ZZZ=QUIT";N$: IF N$="N" THEN GO
 TO 1350
1315 IF N$="ZZZ" THEN RETURN
1320 LET C$=C$+P$(8*(VAL N$-1)+1
 TO 8*VAL N$): IF LEN C$=168 THE
N PRINT AT 21,0;"CHARACTER SET F
ULL": PAUSE  100: RETURN
1330 GO TO 1310
1350 CLS : NEXT I: RETURN
```

Este módulo imprime no *écran* o dicionário e convida o utilizador a especificar quais os caracteres que devem ser incluídos no novo conjunto de caracteres que está a ser criado.

Comentário

Linha 1250. São limpos quaisquer caracteres já existentes definidos pelo utilizador.

Linhas 1270-1350. Estão aqui envolvidos três desvios. O desvio I imprime páginas de vinte caracteres; o desvio J imprime os caracteres individuais que formam as páginas; o desvio K guarda os oito valores que formam cada carácter individual. Cada carácter é enviado para a posição normalmente ocupada por A, na área dos gráficos definidos pelo utilizador, e impresso.

Ensaio do módulo 3.2.4

Deverá agora ser capaz de fabricar novos conjuntos de caracteres, a partir do material fornecido pelo anterior programa.

MÓDULO 3.2.5

```
1360 REM ************************
1370 REM PLACE IN MEMORY
1380 REM ************************
1390 FOR I=1 TO LEN C$: POKE USR
 "A"+I-1,CODE C$(I): NEXT I
1400 FOR I=0 TO 20: PRINT CHR$ (
144+I): NEXT I
1410 INPUT "NAME FOR CHARACTER S
ET?";N$: SAVE N$CODE USR "A",168
1420 INPUT "REWIND THEN ENTER TO
 VERIFY.";Q$: VERIFY N$CODE USR
"A",168: PRINT AT 21,0;"CHARACTE
R SET VERIFIED.": PAUSE 200: RET
URN
```

Este módulo apresenta o conjunto de caracteres, convida o utilizador a dar-lhe um nome e armazena-o como um bloco de código a ser recolhido por outro programa, da forma desejada.

Ensaio do módulo 3.2.5

Deverá agora ser capaz de armazenar os seus conjuntos de caracteres, desligar o *Spectrum* e reintroduzir o novo conjunto de caracteres. Não se esqueça de salvar o programa, em primeiro lugar.

3.3 QUEBRA-CABEÇAS

Este programa apenas exige ser construído de acordo com os próprios desejos do utilizador, permitindo-lhe jogar um antigo jogo de origem chinesa, com os respectivos dois triângulos pequenos, um médio e dois grandes, associados a um quadrado e um paralelogramo. Mais do que isto, no entanto, o programa é uma indicação de como tanto as formas geométricas regulares como irregulares podem ser desenhadas com simplicidade numa variedade de posições ou orientações, sem recorrer a matemáticas complexas, embora seja mais fácil se tiver alguns conhecimentos desta matéria.

MÓDULO 3.3.1

```
1000 REM ************************
1010 REM VARIABLES
1020 REM ************************
1030 DEF FN A( )=SIDE*COS (A*PI/4
)
1040 DEF FN B( )=SIDE*SIN (A*PI/4
)
1050 LET X=100: LET Y=100
```



```
1060 INPUT "CLEAR SCREEN? (Y/N)
";Q$: PAPER 6: INK 1: IF Q$="Y"
THEN CLS
1070 LET L$="MOVE WITH CURSOR AR
ROWS OR "";"""
1080 LET K$="""0"" TO ABANDON SH
APE                "
1090 LET J$="""1"" TO RECORD SCR
EEN OR QUIT. "
1100 DIM Q$(32)
```

Note-se que não há qualquer ementa para este programa. Todas as instruções são dadas ao fundo do *écran*, no decurso do programa. Este módulo organiza as necessárias variáveis.

Comentário

Linhas 1030-1040. Estas duas simples funções são utilizadas para determinar os pontos finais de linhas a desenhar. Para desenhar uma linha utilizando as duas funções, apenas é necessário conhecer o ponto de partida, o ângulo e a extensão da linha. Dados estes, as funções definirão duas coordenadas que serão suficientes para permitir a utilização do comando DRAW. Uma vista de olhos pelas páginas 68 e 69 do manual do *Spectrum* mostrarão como as funções COS e SIN podem ser usadas para conseguir aquele objectivo. Saliente-se que as funções apenas podem tratar os ângulos que podem ser expressos em unidades de PI—4 radianos ou um oitavo de círculo. Se desejar maior flexibilidade, pode ser inserido um divisor maior, tal como 180, o que resultará em unidades de um grau.

Linha 1050. São estes os pontos de partida para qualquer desenho.

MÓDULO 3.3.2

```
1650 REM ************************
1660 REM MOVING PIXEL
1670 REM ************************
1680 PRINT AT 20,0;L$'J$
1690 OVER 1: FOR I=1 TO 2: PLOT
BRIGHT 1 AND I=1;X,Y: PLOT X,Y+1
: PLOT X,Y+2: PAUSE 1+2*(I=1): N
EXT I: OVER 0: PLOT INVERSE 1; O
VER 1;X,Y
1700 LET T$=INKEY$: IF T$="" THE
N GO TO 1690
1710 IF T$="9" THEN RETURN
1720 IF T$="1" THEN GO TO 1330
1730 IF T$>"9" OR T$<"5" THEN GO
 TO 1690
1740 LET X=X+(T$="8")-(T$="5"):
LET X=X-(X>245)+(X<5)
1750 LET Y=Y+(T$="7")-(T$="6"):
LET Y=Y+(Y<20)-(Y>170)
1760 GO TO 1690
```

Este módulo imprime um pequeno cursor intermitente na posição de partida de qualquer forma a desenhar, permitindo ao utilizador mover o cursor sem alterar o conteúdo do *écran*. O método é similar ao do módulo do cursor no primeiro programa deste capítulo, mas aqui apenas estão envolvidos três *pixels*.

Comentário

Linha 1690. Note-se o uso da condição lógica para preparar a característica BRIGHT (brilhante). PLOT INVERSE 1; OVER 1 é uma forma de deslocar a posição do traçado sem traçar um ponto, e é utilizada porque, no final do desvio, a posição do traçado se deslocou para XY+2. A posição do carácter é colocada em BRIGHT, por forma a tornar mais fácil a identificação do pequeno cursor.

Linhas 1740-1750. Estas linhas são exactamente paralelas às que foram usadas no último programa para mover o cursor e delimitar o seu movimento.

Ensaio do módulo 3.3.2

Deverá conseguir passar o programa e movimentar pelo *écran* o reduzido cursor.

MÓDULO 3.3.3

```
1110 REM ************************
1120 REM SELECT SHAPE
1130 REM ************************
1140 GO SUB 1650: PRINT AT 20,0;
0$;K$: LET SUPP=3
1150 INPUT "1:T 2:P 3:S 4:C ";SH
APE: IF SHAPE=0 THEN GO TO 1110
1160 IF SHAPE=4 THEN GO SUB 1410
: GO TO 1110
1170 IF SHAPE=2 THEN INPUT "TYPE
 1 OR 2 ";SUPP: IF SUPP=0 THEN G
O TO 1110
1180 IF SHAPE=1 THEN INPUT "SIZE
 (1,2,3) ";SIZE: IF SIZE=0 THEN
GO TO 1110
1190 INPUT "ORIENTATION ";O: LET
 O=3-O: LET A=O: IF O=3 THEN GO
TO 1110
1200 INPUT "CORNER? ";C1: IF C1=
0 THEN GO TO 1110
1210 LET LONG=36: IF SHAPE=1 THE
N LET LONG=36*((SQR 2)^SIZE)-.5
1220 LET SHORT=LONG/SQR 2
1230 INPUT "DRAW:1 ERASE:2 ";DRA
W: OVER 1
```

```
1240 IF SHAPE=1 THEN GO SUB 1470
1250 IF SHAPE=2 THEN GO SUB 1520
1260 IF SHAPE=3 THEN GO SUB 1570
1270 IF DRAW=1 THEN INPUT "CONFI
RM SHAPE? (Y/N) ";Q$: LET A=0: O
VER (Q$="N"): PLOT INVERSE 1; OV
ER 1;X,Y
1280 IF DRAW<>1 THEN OVER 0: GO
TO 1110
1290 IF SHAPE=1 THEN GO SUB 1470
1300 IF SHAPE=2 THEN GO SUB 1520
1310 IF SHAPE=3 THEN GO SUB 1570
1320 OVER 0: GO TO 1110
```

Este módulo aceita as informações necessárias para desenhar qualquer forma.

Comentário

Linha 1150. O programa, tal como se encontra, é capaz de desenhar quatro formas básicas: triângulos, paralelogramos, quadrados e circulos. Em qualquer inserção neste módulo, «0» resultará no retorno ao cursor intermitente.

Linha 1170. Os paralelogramos não são simétricos, pelo que o programa tem de ser informado sobre qual o caminho a seguir para imprimir um deles. SUPP regista qual dos paralelogramos é especificado.

Linha 1180. Estão disponíveis três tamanhos de triângulos; são simétricos, rectângulos e cada um tem o dobro da área do imediatamente abaixo.

Linha 1190. Para cada forma é escolhido um canto-chave. A seguir, no sentido dos ponteiros do relógio, a partir deste canto, fica o lado-chave, e é o ângulo deste lado que aqui se introduz.

Linha 1200. Tendo determinado o ângulo do lado-chave, fica apenas por dizer qual dos cantos da forma cairá no local marcado pelo cursor. Os cantos encontram-se numerados a partir do canto-chave. Tudo isto lhe pode parecer tremendamente complicado mas, com a utilização, tornar-se-á óbvio.

Linha 1210. Os lados longos dos triângulos pequenos e dos paralelogramos têm 36 *pixels* de extensão, um número escolhido arbitrariamente. Em conformidade, os lados pequenos têm 36/SQR 2 *pixels* de extensão... se discorda, tenha uma conversinha com Pitágoras. Os lados grandes dos triângulos maiores aumentam em múltiplos de 2.

Linhas 1230-1320. Esta secção solicita as rotinas que desenham as formas, excepto o circulo. Cada rotina é solicitada duas vezes. Na primeira ocasião, a forma é desenhada OVER (sobre) aquilo que já lá estiver. Resulta daqui que todas as linhas tocadas serão apagadas ou sobrepostas. Isto continua a possibilitar ao utilizador a faculdade de verificar se a forma está na posição correcta. Se o utilizador confirmar este facto, a forma será desenhada novamente, com OVER em zero, tornando assim válidas quaisquer separações. Se a forma não for confirmada, será redesenhada (OVER), sendo assim apagada. O utilizador pode igualmente apagar uma figura cujo canto (um deles) coincida com o ponto marcado pelo cursor... isto resultará na perda permanente de quaisquer linhas sobrepostas.

Ensaio do módulo 3.3.3

Não podem ainda ser desenhadas quaisquer formas, mas o programa deverá aceitar todas as caracteristicas especificadas em cima.

MÓDULO 3.3.4

```
1410 REM ************************
1420 REM CIRCLE
1430 REM ************************
1440 INPUT "RADIUS? ";R: IF R=0
THEN RETURN
1450 CIRCLE OVER 1;X,Y,R
1460 INPUT "CONFIRM CIRCLE? (Y/N
) ";Q$: CIRCLE OVER (Q$="N");X,Y
,R: RETURN
```

Este módulo permite ao utilizador estabelecer um raio e desenhar um círculo cujas origens serão o ponto marcado pelo cursor, com a opção de o apagar novamente.

Ensaio do módulo 3.3.4

Deverá poder movimentar o cursor até um ponto escolhido, desenhando um círculo em volta desse ponto.

MÓDULO 3.3.5

```
1610 REM ************************
1620 REM DRAW SHAPE
1630 REM ************************
1640 DRAW FN A(),FN B(): RETURN
```

Este módulo deposita o ponto final de uma linha, a qual compreenderá um dos términos de uma forma. Ambas as funções foram já explicadas anteriormente.

MÓDULO 3.3.6

```
1470 REM ************************
1480 REM TRIANGLES
1490 REM ************************
1500 LET A$="036": FOR I=0 TO 2:
 LET I1=C1+I-3*((I+C1)>3): LET S
IDE=SHORT: IF I1=2 THEN LET SIDE
=LONG
1510 LET A=0-VAL A$(I1): GO SUB
1610: NEXT I: RETURN
```

Este módulo, quando utilizado com o anterior, desenha triângulos.

Comentário

Linhas 1500-1510. O método aqui empregue é simples. É criada uma sequência para cada figura. Os caracteres individuais da sequência registam quantas unidades de 45 graus cada linha deve ser rodada, na direcção dos ponteiros do relógio, em relação ao lado-chave. Assim, o lado-chave encontra-se, claramente, nas unidades zero, em relação a si próprio. No caso de um triângulo, o lado contíguo encontra-se a um ângulo de 135 graus, ou três unidades, em relação ao lado-chave. O desvio simples contido nestas duas linhas combina a introdução do ângulo para o lado-chave com cada um dos caracteres da sequência, após ter determinado o ângulo de cada lado. A variável I1 assegura que, se o desvio não começar em 1, este se reduzirá a 1 quando ultrapassar 3. Note-se a divisão no desvio, causada pela presença de um enunciado *If*.

Ensaio do módulo 3.3.6

Deverá agora poder especificar e desenhar triângulos.

MÓDULO 3.3.7

```
1520 REM ************************
1530 REM PARALLELOGRAM
1540 REM ************************
1550 LET A$="0145": FOR I=0 TO 3
: LET I1=C1+I-4*((I+C1)>4): LET
SIDE=LONG: IF I1=2 OR I1=4 THEN
LET SIDE=SHORT
1560 LET A=0-VAL A$(I1)+2*VAL A$
(I1)*(SUPP=2): GO SUB 1610: NEXT
 I: RETURN
```

Este é similar ao módulo 6, excepto por desenhar um paralelogramo. Note-se como SUPP é utilizada para determinar que tipo de paralelogramo é desenhado.

MÓDULO 3.3.8

```
1570 REM ************************
1580 REM SQUARE
1590 REM ************************
1600 LET SIDE=LONG: FOR I=0 TO 3
: LET A=0-2*I: GO SUB 1610: NEXT
 I: RETURN
```

Este módulo imprime no *écran* um quadrado, por um método diferente do utilizado pelos dois anteriores. Neste caso, o ângulo do lado-chave é simplesmente aumentado, de forma regular, desenhando-se um lado. Este método deverá resultar para qualquer figura regular, sendo uma tarefa fácil permitir ao utilizador especificar o número de lados, sendo a alteração de ângulo para cada lado de 2Pi/X, sendo X o número de lados.

Ensaio do módulo 3.3.8

Deverão estar agora disponíveis todas as capacidades do programa para desenhar formas.

MÓDULO 3.3.9

```
1330 REM ************************
1340 REM SAVE PATTERN
1350 REM ************************
1360 INPUT "DO YOU WANT TO QUIT?
 (Y/N)";Q$: IF Q$="Y" THEN STOP
1370 INPUT "NAME OF PATTERN:";N$
1380 PRINT AT 20,0;"RESTART PROG
RAM WITH GO TO 1
                              "
1390 SAVE N$SCREEN$
1400 STOP
```

Este módulo permite que se salve para cassete o modelo criado, para posterior utilização.

Comentário

Linha 1390. Este comando salva para fita o conteúdo de 6912 locais de memória que são usados para armazenar os conteúdos do *écran*. Quando se trabalha com gráficos onde se definiram *pixels* individuais, de preferência a caracteres, este é o único método prático de armazenamento.

Ensaio do módulo 3.3.9

Crie um modelo e salve-o em fita. Limpe o *écran* e solicite de novo o modelo introduzindo LOAD «Nome» SCREEN$ e passando a fita. A imagem deverá reconstruir-se gradualmente. Reiniciando o programa com GOTO 1, deverá poder acrescentar a figura, tal como se ela nunca tivesse estado ausente.

Sumário

Se estiver interessado em produzir formas apenas sob o seu controlo, as técnicas expostas neste capítulo ser-lhe-ão de grande utilidade. Com elas pode imprimir figuras regulares e irregulares tiradas a régua. A técnica para armazenar ângulos relativos numa curta sequência pode ser usada para definir perímetros bastante complexos, que seria praticamente impossível analisar matematicamente. Os módulos para desenhar tais figuras seriam curtos, tal como mostra o programa, requerendo poucas variáveis, pelo que são candidatos práticos à inclusão noutros programas, que poderiam vir a beneficiar com o desenho de algumas formas.

Avançando mais

1) No módulo que desenha círculos não há dispositivo para apagar círculos já desenhados. Poderá acrescentar-lhe um?

2) Experimente acrescentar um módulo que desenhasse um octógono, utilizando o mesmo método que o módulo que desenha quadrados.

3) Altere o programa de forma a que ele possa desenhar em qualquer ângulo, em vez de ficar limitado a unidades de 45 graus.

3.4 ARTISTA

Este programa, ou qualquer outro parecido, é parte importante no processo de construção de uma biblioteca de programas. Se vai possuir programas que utilizem figuras como parte do respectivo visual, então deverá poder criar imagens com simplicidade e armazená-las.

A intenção de qualquer programa deste tipo é realçar as capacidades do *Spectrum*, permitindo mover o cursor por todo o *écran* e possibilitando a impressão de caracteres gráficos em qualquer posição do *écran*, com um mínimo pressionar de teclas. Por último, seria desejável, num tal programa, que os desenhos criados pudessem ser armazenados mais economicamente do que com a utilização da função SCREEN$, a qual requer aproximadamente 7000 *bytes* de memória e é bastante lenta a introduzir. No entanto, uma vez que não vamos utilizar SCREEN$, apenas podemos utilizar caracteres gráficos inteiros, incluindo caracteres definidos pelo utilizador. O programa comportar-se-á como se os desenhos criados pelos comandos DRAW, CIRCLE ou PLOT lá não estivessem.

Nota. — Se tem a intenção de utilizar gráficos definidos pelo utilizador em qualquer desenho destinado a ser armazenado, qualquer programa subsequente que recolha o desenho deverá ser introduzido com os mesmos caracteres.

MÓDULO 3.4.1

```
1000 REM ************************
1010 REM VARIABLES
1020 REM ************************
1030 DIM B$(2,30): FOR I=1 TO 30
: LET B$(1,I)="8": LET B$(2,I)=C
HR$ 0: NEXT I
1040 DIM A$(3,20,30): FOR I=1 TO
 20: LET A$(2,I)=B$(1): LET A$(3
,I)=B$(2): NEXT I
1050 LET X=1: LET Y=1
1060 LET T=4
1070 DIM O$(32)
1080 PAPER 7: INK 0: CLS
1090 LET MODE=1: DIM L$(2,20): L
ET L$(1)="(MODE 1)": LET L$(2)="
(MODE 2)"
1100 GO SUB 1320
```

Comentário

Linha 1030. Esta linha cria um expediente temporário, B$, que é utilizado para apressar o processo de introdução da disposição principal, A$.

Linha 1040. A disposição A$ será utilizada, por fim, para armazenar três conjuntos de informações:

a) Os caracteres correntemente impressos no *écran*.
b) As características cromáticas de cada um dos caracteres.
c) Se o carácter deve ser impresso em vídeo invertido.

A segunda parte da disposição é preenchida com oitos, porque o código deste carácter, 56, é o mesmo de um *byte* atributo determinando uma tinta 0 e um papel 7. Para mais explicações sobre os códigos ATTR, que determinam as características cromáticas, ver a p. 116 do manual do *Spectrum*. A terceira secção da disposição é preenchida com CHR$ 0, o que significa a inexistência de caracteres invertidos.

Linha 1060. Esta variável regista a direcção em que se movimentará o cursor, expressa em segmentos de um oitavo de círculo, começando com 1 na vertical.

Linha 1090. MODE é a variável utilizada para registar a impressão de caracteres normais ou invertidos.

MÓDULO 3.4.2

```
1320 REM ************************
1325 REM ACCEPT CHARACTERS
1330 REM ************************
1340 PRINT INK 0; PAPER 7;AT 0,0
;L$(MODE);AT 21,0;"SPACE W=ATTR
X=ST Y=REC Z=MO  ";AT 0,21;"DIRE
CTION=";T
1350 PRINT INK 0; PAPER 7; OVER
1;AT X,Y;"*": PAUSE 3: PRINT INK
 0; PAPER 7; OVER 1;AT X,Y;"*"
1360 LET T$=INKEY$
1370 IF T$="" THEN GO TO 1350
1380 BEEP .05,12
```



```
1390 IF CODE T$>32 AND CODE T$<4
1 THEN LET T=CODE T$-32: GO TO 1
340
1400 IF CODE T$=64 THEN LET T=2:
 GO TO 1340
1410 IF T$="W" THEN GO SUB 1170:
 GO TO 1320
1420 IF T$="X" THEN STOP
1430 IF T$="Y" THEN GO SUB 1600:
 GO TO 1320
1440 IF T$="Z" THEN LET MODE=MOD
E+1: LET MODE=MODE-3*(MODE>2): G
O TO 1320
1450 IF T$=" " THEN POKE (22528+
32*X+Y),CODE A$(2,X,Y): GO SUB 1
540: GO TO 1340
1460 IF T$="0" THEN PRINT AT X,Y
;" ": LET A$(1,X,Y)=" ": LET A$(
2,X,Y)=CHR$ ATTR (X,Y): GO SUB 1
540: GO TO 1340

1470 IF MODE<>1 THEN GO TO 1510
1480 IF CODE T$>=49 AND CODE T$<
=56 THEN LET T$=CHR$ (CODE T$+80
): LET T$=CHR$ (CODE T$-8*(CODE
T$=136)): GO SUB 1110: LET A$(3,
X,Y)=CHR$ 0: GO SUB 1540: GO TO
1350
1490 IF CODE T$>=65 AND CODE T$<
=85 THEN LET T$=CHR$ (79+CODE T$
): LET A$(3,X,Y)=CHR$ 0: GO SUB
1110: GO SUB 1540: GO TO 1350
1500 IF MODE<>2 THEN GO TO 1350
1510 IF CODE T$>=49 AND CODE T$<
=56 THEN LET T$=CHR$ (191-CODE T
$): LET T$=CHR$ (CODE T$+8*(CODE
 T$=135)): LET A$(3,X,Y)=CHR$ 0:
 GO SUB 1110: GO SUB 1540: GO TO
 1350
```

```
1520 IF CODE T$>=65 AND CODE T$<
=85 THEN LET T$=CHR$ (79+CODE T$
): LET A$(3,X,Y)=CHR$ 1: GO SUB
1110: GO SUB 1540: GO TO 1350
1530 GO TO 1320
```

Este módulo distribui trabalho por entre os módulos que constituem o resto do programa e transfere a inserção de caracteres para caracteres gráficos, de acordo com o modo em vigor.

Comentário

Linhas 1390-1400. Não se pede ao utilizador que especifique a direcção ou desloque o cursor de cada vez que é introduzido um carácter. Ele move-se automaticamente numa de oito direcções. A direcção é determinada premindo a tecla *Symbol Shift* continuamente e premindo uma das teclas de 1 a 8.

Linha 1450. Se o utilizador deseja deslocar o cursor por sobre um carácter sem alterar as características cromáticas daqueles que estiverem em acção, isto pode simplesmente ser feito pela introdução de um espaço. As características cromáticas originais da posição daquele carácter, que foram armazenadas na segunda parte de **A$**, são devolvidas ao *byte* dos atributos correspondente ao espaço do carácter. Terá notado já, espero, que duas impressões do cursor utilizando OVER asseguram que o carácter corrente, no respectivo espaço, ficará inalterável, de qualquer forma.

Linha 1460. A inserção de zero apaga o que quer que esteja no quadro do carácter.

Linhas 1480-1520. De acordo com o modo estabelecido, o código corrente do carácter da tecla premida é transferido para o código do carácter gráfico desejado. A rotina que imprime o carácter é então solicitada e o cursor deslocado. Note-se que os caracteres invertidos associados às teclas 1 a 8 não estão realmente invertidos. São apenas outro conjunto de caracteres tirado do conjunto de caracteres. Os caracteres invertidos definidos pelo utilizador estão realmente invertidos. Para estes, é colocado um indicador na terceira área de **A$**.

Ensaio do módulo 3.4.2

Com este módulo instalado, deverá ver o cursor intermitente, as instruções abreviadas ao fundo do *écran* e a direcção do cursor no canto superior direito. Deverá poder alterar a direcção mantendo a tecla *Symbol Shift* premida e carregando numa das teclas 1 a 8. Poderá mudar o modo premindo a tecla Z. Não pode ainda introduzir quaisquer caracteres.

MÓDULO 3.4.3

```
1540 REM ************************
1550 REM MOVE INCREMENT
1560 REM ************************
1570 LET X=X+(T=4)+(T=5)+(T=6)-(
T=8)-(T=1)-(T=2): LET X=X+(X<1)-
(X>20)
1580 LET Y=Y-(T=6)-(T=7)-(T=8)+(
T=2)+(T=3)+(T=4): LET Y=Y+(Y<1)-
(Y>30)
1590 RETURN
```

Este é um módulo padrão de movimentação do cursor.

Ensaio do módulo 3.4.2

Poderá agora movimentar o cursor na direcção desejada, premindo as teclas de espaço ou zero.

MÓDULO 3.4.4

```
1110 REM ************************
1120 REM ORDINARY GRAPHICS
1130 REM ************************
1140 LET A$(1,X,Y)=T$
1150 PRINT INVERSE (CODE A$(3,X,
Y)=1);AT X,Y;T$
1160 LET A$(2,X,Y)=CHR$ ATTR (X,
Y): RETURN
```

Este módulo imprime o carácter desejado na posição do cursor.

Comentário

Linha 1140. O carácter a ser impresso é colocado na posição equivalente, na primeira parte de **A$**.

Linha 1150. O carácter é impresso em inverso, se houver um indicador na terceira secção de **A$**.

Linha 1160. As características cromáticas são recolhidas utilizando a função ATTR, e armazenadas na segunda área de **A$**.

Ensaio do módulo 3.4.4

Deverá agora poder imprimir quaisquer caracteres gráficos, em qualquer posição, dentro da grelha de 20*30. Não pode determinar características cromáticas até à introdução do próximo módulo.

MÓDULO 3.4.5

```
1170 REM ************************
1180 REM ATTRIBUTES
1190 REM ************************
1200 PRINT AT 0,0;"ATTRIBUTES
"
1210 DIM Q(4)
1220 FOR I=1 TO 4
1230 INPUT CHR$ (216+I);Q(I)
1240 IF Q(I)>9 THEN GO TO 1230
1250 IF I=1 THEN INK Q(I)
1260 IF I=2 THEN PAPER Q(I)
1270 IF I>2 AND Q(I)>1 THEN GO T
O 1230
1280 IF I=3 THEN FLASH Q(I)
1290 IF I=4 THEN BRIGHT Q(I)
1300 NEXT I
1310 RETURN
```

Este módulo simples pede quatro entradas, que vão determinar as características de INK (tinta), PAPER (papel), FLASH (intermitência) e BRIGHT (brilho).

Ensaio do módulo 3.4.5

Deverá agora poder variar as características cromáticas em vários pontos do *écran*.

MÓDULO 3.4.6

```
1600 REM ************************
1610 REM CREATE PACKED STRING
1620 REM ************************
1630 PRINT INK 0; PAPER 7;AT 21,
```

```
0;"RECORDING
   ": INVERSE 0
1640 LET S$="": LET SPACE=0: LET
 IND=1
1650 FOR I=1 TO 20: FOR J=1 TO 3
0
1660 PRINT INK 0;AT I,0;"_"
1670 IF A$(1,I,J)=" " THEN LET S
PACE=SPACE+1: LET IND=1: GO TO 1
700
1680 IF IND=1 THEN LET S$=S$+CHR
$ 255+CHR$ SPACE: LET IND=0: LET
 SPACE=0
1690 LET S$=S$+A$(1,I,J)+A$(2,I,
J)+A$(3,I,J)
1700 NEXT J
1710 IF IND=1 THEN LET S$=S$+CHR
$ 255+CHR$ SPACE
1720 LET IND=1: LET SPACE=0
1730 NEXT I
1740 BEEP 1,2
```

Este módulo cria uma sequência compacta, ou seja, uma sequência com todos os espaços removidos, segundo o modelo armazenado em **A$**. O número de espaços removidos é armazenado em marcadores de carácter único, dentro da sequência.

Comentário

Linha 1670. Se o desvio encontra um espaço, regista-o na variável SPACE e prepara o marcador IND. Este processo continua até ser encontrado um carácter não espaço.

Linha 1680. Tendo encontrado um carácter que não é um espaço, é acrescentado um indicador à sequência (CHR$ 255 ou COPY), seguido de um carácter único que regista o número de espaços que foram deixados fora.

Linha 1690. Os três *bytes* relevantes das três áreas de **A$** são acrescentados à sequência compacta.

Linha 1710. O fim de cada linha de **A$** é marcado com COPY, seguido do número de quaisquer espaços que precedessem o fim da linha.

MÓDULO 3.4.7

```
1750 REM ************************
1760 REM PRINT PACKED STRING
1770 REM ************************
1780 PAPER 7: CLS : PRINT AT 1,1
;: LET S=0: LET C=1
1790 IF S$(C)=CHR$ 255 THEN LET
C=C+1: IF S$(C)<>CHR$ 0 THEN PRI
NT O$( TO CODE S$(C));: LET S=S+
CODE S$(C): PRINT "  " AND S/30=
INT (S/30);: LET C=C+1: IF C<=LE
N S$ THEN GO TO 1790
1800 IF C>LEN S$ THEN GO TO 1840
1810 IF S$(C)=CHR$ 0 THEN LET C=
C+1
1820 LET S=S+1: INVERSE CODE S$(
C+2): PRINT S$(C);: POKE (23296-
32*(PEEK 23689)-PEEK 23688+32),C
ODE S$(C+1): INVERSE 0: IF S/30=
INT (S/30) THEN PRINT "  ";
1830 LET C=C+3: IF C<=LEN S$ THE
N GO TO 1790
1840 INPUT INK 0; PAPER 7;"SAVE
IT? (Y/N)";0$
1850 IF 0$<>"Y" THEN RETURN
1860 INPUT "PLEASE GIVE THE PICT
URE A NAME:";N$
1870 DIM P$(LEN S$): LET P$=S$
```

```
1880 SAVE N$ DATA P$( )
1890 INPUT "REWIND TAPE, PRESS "
"ENTER""    THEN START TAPE TO V
ERIFY";Q$
1900 VERIFY N$ DATA P$( )
1910 PRINT AT 21,0;"PICTURE VERI
FIED:": STOP
1920 INPUT T$: PRINT CODE T$: GO
 TO 1920
```

Este módulo reimprime a sequência compacta, para inspecção, e salva-a a pedido.

Comentário

Linha 1790. Esta linha examina a sequência compacta, em busca do indicador especial, COPY, que indica que o próximo carácter regista um número de espaços que foram removidos. Quando é encontrado um marcador, é inserido o número correcto de espaços. Note-se que têm de ser impressos dois espaços no fim de cada linha, tendo em conta que a grelha original tinha apenas 30 caracteres de extensão.

Linha 1820. Se o carácter na posição C não for um marcador será impresso, invertido se o segundo carácter a seguir, C+2, for CHR$ 1. As caracteristicas cromáticas são então estabelecidas através da introdução (POKE) do valor do carácter C+1 na área dos atributos.

Linha 1830. O contador avança três espaços, pois cada carácter necessita de três espaços na sequência compacta.

Linhas 1840-1910. Se o utilizador deseja salvar o desenho, tal como se encontra armazenado na sequência compacta, àquele será primeiro dado um nome. Note-se que o *Spectrum* não salvará sequências correctamente, se estas não tiverem sido dimensionadas, pelo que temos de criar uma nova sequência, P$, do mesmo comprimento que a sequência compacta.

Uma vez salvo, o desenho pode ser recolhido da fita por um outro programa e reimpresso por uma rotina como a das linhas 1780-1830. Ver igualmente o módulo 4.2.5.

Sumário

Utilizando este grupo de três programas, terá um conjunto de ferramentas que lhe permitirão melhorar muitos dos seus programas e intentar muitas aplicações que não seriam praticáveis sem eficientes ferramentas gráficas.

Ao desenhar os seus próprios gráficos poderá achar úteis algumas folhas de papel quadriculado, como ajuda na esquematização das suas intenções, antes de se sentar em frente ao teclado; verificará que economiza bastante tempo. Se conseguir encontrar desenhos com as dimensões adequadas, isso também poderá ajudar a passá-los para plástico transparente, colocando depois o plástico sobre o *écran* televisivo. Depois, bastará seguir o modelo traçado.

Avançando mais

1) Um desenho a todo o *écran* armazenado por este programa pode, apesar de tudo, requerer até 180 caracteres. Se necessitar das características de inversão nos seus desenhos, este número poderá ser reduzido em um terço. Se só necessitar de desenhos de cor única, apenas precisará do espaço de um só carácter para gravar cada carácter do desenho. Numa das aplicações, no capítulo educacional, assume-se que você modificou os módulos da sequência compacta, por forma a que eles armazenem e reproduzam apenas os caracteres e não os atributos.

2) Modele um dicionário para os seus desenhos, se tiver memória. Será um programa que armazenará um número razoável de desenhos e os passará a pedido ou em sequência.

3) Podem ser acrescentados mais dois modos, um para aceitar letras e números do teclado normal e o outro para aceitar a inversão daqueles. O problema é que as ordens não podem ser dadas, a menos que comece a excluir teclas. Utilizando ENTER (CHR$ 13) como comando para mudar de modo, pode usar os dois modos, embora não comandos de saída. Poderá adaptar o programa para isto?

3.5 DESENHISTA

Tenho uma afeição especial por este programa, porque as ideias em que ele se baseia não partiram de mim: foram retiradas de um excelente livro, *The Principles of Interactive Cumputer Graphics*, de William M. Newman e Robert F. Sproull. O motivo por que me referi a afeição justifica-se porque o programa me serve de lembrança de como há sempre muito a aprender acerca dos princípios de programação e de quantos campos permanecem por explorar, a custo pouco superior ao da compra de alguns livros. Baseado em simples processos retirados do livro, este programa permite-lhe definir um desenho até 65536*65536 *pixels*, acrescentar e apagar, examinar o desenho a várias escalas e rodá-lo, no todo ou em parte, dentro dos limites do *écran*. Uma vez dominada a sua utilização, ele é capaz de ser utilizado numa variedade de aplicações onde é desejável que se possa alterar e manipular desenhos rápida e facilmente.

MÓDULO 3.5.1

```
1000 REM ************************
1010 REM MENU
1020 REM ************************
1030 INK 0: PAPER 6: CLS : PRINT
 PAPER 2; INK 7;AT 1,10;"DESIGNE
R"
1040 PRINT '"COMMANDS AVAILABLE:
"
1050 PRINT "     1)INITIIALISE DI
SPLAY"
1060 PRINT "     2)ADD NEW LINES"
1070 PRINT "     3)SCALE/ROTATE"
1080 PRINT "     4)DELETE LINES"
1090 PRINT "     5)STOP"
1100 INPUT Z$: CLS
1110 IF Z$="1" THEN GO SUB 1190
```



```
1120 IF Z$="2" THEN GO SUB 1530
1130 IF Z$="3" THEN LET SEARCH=0
: GO SUB 1780
1140 IF Z$="4" THEN LET SEARCH=1
: GO SUB 1780
1150 IF Z$="5" THEN GO TO 1170
1160 CLS : GO TO 1000
1170 INPUT "DO YOU WISH TO SAVE
THIS DESIGN? ";Q$: IF Q$="Y" THE
N SAVE "DESIGNER": PRINT "REWIND
,THEN ANY KEY TO VERIFY": PAUSE
0: VERIFY "DESIGNER": PRINT "VER
IFIED"
1180 STOP
```

Trata-se de um módulo de ementa padrão.

MÓDULO 3.5.2

```
1190 REM ************************
1200 REM INITIALISE
1210 REM ************************
1220 LET LEFT=0: LET BOTTOM=0: L
ET TOP=167: LET RIGHT=255
1230 LET A$=""
1240 DEF FN A( )=256*CODE (A$( I1
))+CODE A$( I1+1)
1250 DEF FN A$( )=CHR$ INT (TX1/2
56)+CHR$ (TX1-256*INT (TX1/256))
+CHR INT (TY1/256)+CHR$ (TY1-25
6*INT (TY1/256))
1260 DEF FN B$( )=CHR$ INT (TX2/2
56)+CHR$ (TX2-256*INT (TX2/256))
+CHR$ INT (TY2/256)+CHR$ (TY2-25
6*INT (TY2/256))
1270 LET A$=" ": RETURN
```

As funções e variáveis aqui definidas serão discutidas no decurso do comentário ao programa.

MÓDULO 3.5.3

```
1280 REM ************************
1290 REM DRAW LINES
1300 REM ************************
1310 IF (X1<LEFT AND X2<LEFT) OR
 (X1>RIGHT AND Y2>RIGHT) OR (Y1>
TOP AND Y2>TOP) OR (Y1<BOTTOM AN
D Y2<BOTTOM) THEN LET OUT=1: RET
URN
1320 IF Y1>TOP THEN LET EDGE=TOP
1330 IF Y1<BOTTOM THEN LET EDGE=
BOTTOM
1340 IF Y1<BOTTOM OR Y1>TOP THEN
 LET X1=X1+(X2-X1)*(EDGE-Y1)/(Y2
-Y1): LET Y1=EDGE
1350 IF Y2>TOP THEN LET EDGE=TOP
1360 IF Y2<BOTTOM THEN LET EDGE=
BOTTOM
1370 IF Y2<BOTTOM OR Y2>TOP THEN
 LET X2=X2+(X1-X2)*(EDGE-Y2)/(Y1
-Y2): LET Y2=EDGE
1380 IF X1>RIGHT THEN LET EDGE=R
IGHT
1390 IF X1<LEFT THEN LET EDGE=LE
FT
1400 IF X1<LEFT OR X1>RIGHT THEN
 LET Y1=Y1+(Y2-Y1)*(EDGE-X1)/(X2
-X1): LET X1=EDGE
1410 IF X2>RIGHT THEN LET EDGE=R
IGHT
1420 IF X2<LEFT THEN LET EDGE=LE
FT
```



```
1430 IF X2<LEFT OR X2>RIGHT THEN
 LET Y2=Y2+(Y1-Y2)*(EDGE-X2)/(X1
-X2): LET X2=EDGE
1440 IF X1-LEFT>=0 AND X2-LEFT>=
0 AND X1-LEFT<=255 AND X2-LEFT<=
255 AND Y1-BOTTOM>=0 AND Y1-BOTT
OM<=167 AND Y2-BOTTOM>=0 AND Y2-
BOTTOM<=167 THEN PLOT X1-LEFT,Y1
-BOTTOM+8: DRAW INT (X2-X1),INT
(Y2-Y1)
1450 RETURN
```

A função deste módulo é tirar dois conjuntos de coordenadas, X1/Y1 e X2/Y2, e decidir se alguma parte de uma linha desenhada entre os dois pontos assim definidos passará pelo *écran*. Se assim acontecer efectivamente com alguma parte da linha, será desenhada, sendo rejeitado o resto da linha.

Comentário

Linha 1310. O *écran* forma uma janela sobre o conjunto do desenho em criação e as margens da área para onde aponta o *écran* são armazenadas nas variáveis TOP, BOTTOM, LEFT e RIGHT. Se BOTTOM (fundo) for colocado em 500 e LEFT (esquerda) em 500, o *écran* ficará pronto a passar quaisquer *pixels* entre 500 e 755 horizontais, e 500 e 667 verticais. O objectivo desta linha progamática é o de não ter em consideração qualquer linha do desenho que comece e acabe acima, abaixo, ou para qualquer dos lados da área do desenho coberta pelo *écran*.

Linhas 1320-1330. Se uma linha começar acima ou abaixo da área coberta pelo *écran*, estas duas linhas repõem a variável EDGE (limite) a coincidir com o topo ou fundo do *écran*.

Linha 1340. Para linhas que comecem acima ou abaixo do *écran*, esta linha calcula a posição horizontal em que a linha passará o limite superior ou inferior. A fórmula na primeira metade da linha nada de mais complexo diz do que isto. Se, por exemplo, a li-

nha em questão passar pelo limite superior do *écran*, a meio da sua componente vertical, passará igualmente a meio da sua componente horizontal. É claro que isto apenas será verdadeiro para linhas rectas.

Linhas 1350-1430. É adoptado o mesmo processo em relação às coordenadas Y1, X2 e Y2.

Linha 1440. Uma vez que é possivel a uma linha não permanecer completamente acima, abaixo ou para um dos lados do *écran*, e ainda assim não atravessar o próprio *écran*, esta linha programática verifica se as coordenadas calculadas estão, de facto, dentro dos limites do *écran*. Se assim acontecer, então o primeiro conjunto de coordenadas será traçado e limitado o segundo conjunto.

Ensaio do módulo 3.5.3

Se este módulo estiver a funcionar correctamente, deverá poder correr o programa e iniciar as variáveis. Tendo feito isto, pare o programa e introduza, em modo directo, 127/200 e 127/100 como valores para X1/Y1 e X2/Y2. Estes quatro valores deverão resultar, se você passar à 1280 (GOTO 1280), no desenho de uma linha a partir do meio do limite superior do *écran* até, aproximadamente, o meio deste.

MÓDULO 3.5.4

```
1530 REM ************************
1540 REM CURSOR
1550 REM ************************
1560 LET ANGLE=0: LET S=1: GO SU
B 1820
1570 PRINT AT 21,18;"0QUIT1DEF2M
OVE";AT 21,0;"X1=";01;AT 21,8;"Y
1=";02: LET X=01: LET Y=02
1580 GO SUB 1660: IF T$="2" THEN
 GO SUB 1820: PRINT AT 21,18;"0Q
UIT1DEF2MOVE": LET X=01: LET Y=0
2: GO TO 1570
1590 LET TTX1=X: LET TTY1=Y: IF
T$="0" THEN RETURN
1600 PRINT AT 21,0;"X2=";X;"  ";
AT 21,8;"Y2=";Y;"  ": PAUSE 50:
GO SUB 1660: IF T$="2" THEN GO S
UB 1820: PRINT AT 21,18;"0QUIT1D
EF2MOVE": LET X=01: LET Y=02: GO
 TO 1600
1610 IF T$="0" THEN RETURN
1620 LET X2=X: LET TX2=X: LET Y2
=Y: LET TY2=Y: LET X1=TTX1: LET
Y1=TTY1: LET TX1=X1: LET TY1=Y1:
 OVER 1: GO SUB 1280
1630 INPUT "OK? ";Q$: IF Q$="Y"
THEN OVER 0: LET A$=A$+FN A$()+F
N B$()
1640 GO SUB 1280: OVER 0: PRINT
AT 21,0;"X1";AT 21,8;"Y1": IF Q$
<>"Y" THEN PLOT OVER 1;TX1-LEFT,
TY1-BOTTOM+8: PLOT OVER 1;TX2-LE
FT,TY2-BOTTOM+8
1650 GO TO 1580
1660 PLOT BRIGHT 1; OVER 1;X-LEF
T,Y-BOTTOM+8: PAUSE 2: PLOT BRIG
HT 0; OVER 1;X-LEFT,Y-BOTTOM+8:
LET T$=INKEY$: IF T$="" THEN GO
TO 1660
1670 LET X=X+(T$="8")-(T$="5")
1680 LET X=X+10*(T$="I")-10*(T$=
"T")
1690 LET Y=Y+(T$="7")-(T$="6")
1700 LET Y=Y+10*(T$="U")-10*(T$=
"Y")
1710 IF X>RIGHT THEN LET X=RIGHT
1720 IF X<LEFT THEN LET X=LEFT
1730 IF Y>TOP THEN LET Y=TOP
1740 IF Y<BOTTOM THEN LET Y=BOTT
```

```
OM
1750 IF T$="1" THEN PLOT OVER 1;
X-LEFT,Y-BOTTOM+S: RETURN
1755 IF T$="0" OR T$="2" THEN RE
TURN
1760 PRINT AT 21,3;X;"  ";AT 21,
11;Y;"  "
1770 GO TO 1660
```

O objectivo deste módulo é permitir ao utilizador movimentar um pequeno cursor pelo *écran*, de forma a estabelecer as coordenadas iniciais e finais de uma linha.

Comentário

Linha 1560. Embora o programa seja capaz de encolher o conjunto do desenho através de qualquer factor especificado e de rodá-lo, apenas podem ser introduzidas linhas com o desenho em dimensão normal e sem ter sido rodado. Esta linha fixa o ângulo do desenho (ANGLE) e o factor de redução (S) em conformidade, antes de solicitar o módulo que desenha a parte do desenho que o utilizador tem interesse em ver no *écran*,

Linha 1570. Deverá ter notado que o *écran* tem apenas 168 *pixels* de altura, em vez dos 176 permitidos. Isto possibilita a utilização da linha 21 para mostrar as coordenadas correntes do cursor, em relação ao canto inferior esquerdo do conjunto do desenho.

Linha 1590. No decurso do programa são utilizados cinco conjuntos de variáveis, em vários pontos, para armazenar os mesmos dados, nomeadamente X/Y, X1/Y1, X2/Y2, TX1/TY1 e TTX1/TTY2. Isto deve-se ao simples facto de que, por vezes, o valor de uma coordenada é alterado por qualquer motivo temporário. O T inicial é uma indicação de que isto apenas se destina a um local de armazenamento temporário. Esta linha permite igualmente ao utilizador movimentar a janela aberta pelos limites do *écran*, após definir o início de uma linha. Nestas linhas transitórias, pode definir-se tudo o que passe além da área de um só *écran*.

Linha 1630. Tendo desenhado a linha especificada, pede-se ao utilizador que a confirme ou não. Se ela for confirmada, armazenam-se as duas coordenadas, X1/Y1 e X2/Y2, sob a forma de dois *bytes*, criados por FN A$ e FN B$, na sequência indefinida A$.

Linhas 1660-1770. Reconhecerá aqui uma rotina padrão de movimentação de um cursor. Existe, no entanto, uma diferença: para além das próprias teclas cursoras, as teclas imediatamente abaixo delas e para a direita (TYUI) podem ser utilizadas para mover o cursor de *pixel* dez lugares de uma só vez, acelerando assim o processo. Premindo 1 provoca-se o retorno a uma parte anterior do módulo, definindo assim um dos conjuntos de coordenadas.

Ensaio do módulo 3.5.4

Introduzindo uma linha temporária 1820 com Retrocesso (Return) e definindo 01 e 02, em modo directo, como 127 e 83 respectivamente, poderá solicitar este módulo e movimentar o cursor pelo *écran*, definir duas posições — não pode ainda mover o *écran* dentro dos limites da linha — e ver a linha mostrada, para que a possa confirmar ou não.

MÓDULO 3.5.5

```
1780 REM **********************
1790 REM ROTATE
1800 REM **********************
1810 INPUT "ANGLE?",ANGLE: LET A
NGLE=ANGLE*PI/180: INPUT "SCALE
FACTOR?",S: IF S=0 THEN LET S=1
1820 INPUT "CO-ORDINATES OF ORIG
IN?";01,02: IF Z$="2" AND 01<127
 THEN LET 01=127
```

```
1830 IF Z$="2" AND O2<83 THEN LE
T O2=83
1840 LET LEFT=O1-127: LET BOTTOM
=O2-83: LET RIGHT=LEFT+255: LET
TOP=BOTTOM+167
1850 CLS
1860 FOR I=2 TO LEN A$ STEP 8
1870 IF I>LEN A$ THEN LET SEARCH
=0: GO TO 1970
1880 LET I1=I: LET X1=(FN A()-O1
)/S: LET I1=I1+2: LET Y1=(FN A()
-O2)/S: LET I1=I1+2: LET X2=(FN
A()-O1)/S: LET I1=I1+2: LET Y2=(
FN A()-O2)/S
1890 LET TX1=X1: LET TX2=X2: LET
 TY1=Y1: LET TY2=Y2
1900 LET X1=O1+INT (TX1*COS ANGL
E+TY1*SIN ANGLE)
1910 LET X2=O1+INT (TX2*COS ANGL
E+TY2*SIN ANGLE)
1920 LET Y1=O2+INT (-TX1*SIN ANG
LE+TY1*COS ANGLE)
1930 LET Y2=O2+INT (-TX2*SIN ANG
LE+TY2*COS ANGLE)
1940 IF SEARCH=1 THEN OVER 1
1950 GO SUB 1280: IF SEARCH=1 TH
EN GO SUB 1460: IF T$="0" THEN L
ET T$="": LET SEARCH=0: RETURN
1960 NEXT I: LET SEARCH=0
1970 INPUT "ENTER=CONTINUE/""CCC
=COPY";Q$
1980 IF Q$="CCC" THEN COPY
1990 RETURN
```

É este o módulo que permite a movimentação da janela representada pelo *écran* segundo o comando de desenho.

Comentário

Linha 1820. A posição do *écran* relativamente ao conjunto do desenho é definida pela marcação da posição do centro do *écran*. Note-se igualmente que, quando a função do programa é 2, apenas ficam disponíveis para o utilizador posições com coordenadas positivas. Isto significa que o utilizador apenas pode desenhar linhas em partes do desenho que tenham endereços positivos. Isto justifica-se pelo facto de os endereços negativos não poderem ser armazenados em A$ pelas duas funções FN A$ e FN B$. Em outras ocasiões durante a execução do programa, tais como quando um desenho é rodado, podem criar-se linhas cujos extremos tenham coordenadas negativas, sendo estas impressas sem problemas, se a janela do *écran* estiver a apontar para elas.

Linha 1840. As margens do *écran* são dispostas de modo a condizerem com o centro especificado do *écran*.

Linhas 1860-1880. Utilizando a função FN A, os valores armazenados em A$ são transferidos de volta a coordenadas numéricas. São transferidos para posições relativas ao centro da janela *(écran)*. Esta distância é então multiplicada pelo factor de escala. As coordenadas são então rodadas em torno do centro do écran, segundo o ângulo requerido.

Linhas 1890-1930. O processo da movimentação de um ponto com as coordenadas X e Y, segundo um determinado ângulo, digamos A, é a aplicação da fórmula: X2 = X*COS A + Y*SIN A e Y1 = —X*SIN A + Y*COS A.

Linha 1940. A variável SEARCH (busca) é utilizada para indicar que o módulo de apagamento deve ser solicitado.

Ensaio do módulo 3.5.5

Poderá agora mover a janela sobre o seu desenho e mover igualmente o *écran* entre o primeiro e segundo conjuntos de coordenadas, ao definir uma linha. Poderá igualmente imprimir o desenho completo, ou parte dele, em várias escalas e em vários ângulos.

```
1460 REM **********************
1470 REM DELETE LINES
1480 REM **********************
1490 PRINT OVER 0;AT 21,0;"YCONF
IRMNDELETE0QUIT": LET OUT=0: OVE
R 1: GO SUB 1280: GO SUB 1280: L
ET T$=INKEY$: IF OUT=1 THEN GO T
O 1510
1500 IF T$<>"0" AND T$<>"Y" AND
T$<>"N" THEN GO TO 1490
1510 OVER 0: IF T$="Y" OR OUT=1
THEN GO SUB 1280: RETURN
1520 OVER 1: GO SUB 1280: PAUSE
50: OVER 0: LET A$=A$( TO I-1 )+A
$( I+8 TO ): LET I=I-8: RETURN
```

Este módulo desenha a linha apontada pela variável-desvio I, no módulo anterior. A linha é desenhada duas vezes com OVER, sendo então dada uma oportunidade ao utilizador de dizer se a linha permanece, ou se o programa deixa o módulo, ou se o endereço da linha relevante é removido de A$. A linha ficará intermitente até ser introduzida uma destas informações.

Ensaio do módulo 3.5.6

Poderá agora apagar linhas.

Sumário

Este programa é uma ferramenta útil em variadas circunstâncias, se houver alguma imaginação. Podem planificar-se esboços, desenhar mapas ou, simplesmente, fazer garatujas. De facto, pode simular muitas das capacidades de computadores de gráficos mais caros, amados por cientistas e engenheiros em muitos campos. Mas não se esqueça que o programa é também um exemplo de uma técnica de fácil aquisição, aplicada ao *Spectrum*. Os livros estão ai para todos nós, a abarrotar de poderosas ideias que ajudam a libertar o poder do seu micro.

Avanço mais

1) Conseguiria combinar este programa com as técnicas de desenho de formas encontradas no programa «Quebra-cabeças», equipando o endereço inicial com certas formas comuns, a serem especificadas em A$?

2) O programa seria mais flexível se previsse a impressão de textos, como parte do conjunto do desenho. Uma vez mais, seria necessário armazenar as coordenadas em A$.

CAPÍTULO 4

FACILITAR A APRENDIZAGEM — O «SPECTRUM» COMO PROFESSOR PARTICULAR

Neste capítulo iremos abordar três programas que permitem ao *Spectrum* ser aplicado no campo da educação caseira. O primeiro destes, «MultiQ», é um programa concebido para permitir ao utilizador a introdução de uma série de questões e respostas, que são depois usadas como base para testes de escolha múltipla criados sem uma ordem estabelecida. O segundo programa é «Palavras», um orientador de leitura básica. Por fim, «Onde?», ensina-lhe a localização de cidades de qualquer país programado.

O propósito destes programas é dar uma ideia do que pode ser conseguido sem demasiado esforço, na área da educação em casa. Ainda assim, a menos que pretenda comprar uma gama de *software* em cassete, com programas especializados dedicados a assuntos individuais, conjuntamente com os seus próprios ficheiros de dados, a utilidade dos seus programas educativos dependerá sempre da quantidade de trabalho que está disposto a dedicar-lhes. O melhor programa de perguntas de escolha múltipla do mundo de pouco servirá, a menos que você esteja interessado em introduzir-lhe o número suficiente de perguntas para o tornar interessante.

Se está preparado para ter algum trabalho em dotar estes programas dos dados necessários para trabalhar, eles podem oferecer-lhe, com frequência, resultados espectaculares, pela simples razão de que eles trabalham ao ritmo da pessoa que com eles quer aprender, não mostrando quaisquer sinais de impaciência quando a evolução é lenta e estando sempre prontos para tentar de novo.

Este programa é um dos meus favoritos. Quando o escrevi fiquei satisfeito por ver que era uma obra competente e que executaria a tarefa para que fora desenhado. Mas só depois de ter inserido uma massa de perguntas e respostas, e de o ter experimentado com outras pessoas, é que me apercebi de que tais programas tornam o ensino tão aditivo como qualquer jogo.

De forma idêntica ao «Unificha», este programa é um camaleão, concebido para mudar de cor segundo as necessidades do utilizador. Em certa altura, você pode desejar que ele seja um explicador de francês, oferecendo uma variedade de palavras como possíveis traduções de uma única palavra inglesa. Algum tempo depois, pode o programa estar a fazer perguntas algo complexas sobre a história do século XIX, dando uma série de datas como respostas possíveis. O objectivo do programa é permitir-lhe fazer isto e muito mais sem que tenha de fazer alterações ao próprio programa.

O programa é também interessante pela maneira como armazena os dados. O «Unificha», como devem lembrar-se, guardava os seus dados numa única ficha comprida, mantendo-se a par de cada acesso por meio de uma disposição para indicadores. O «MultiQ» armazena os seus dados numa série de disposições bidimensionais. Por exemplo, A$ (1000,10) forneceria 1000 espaços com 10 caracteres de comprimento. Isto tem desvantagens, como a de tornar impossível a utilização do espaço ao máximo, como já antes foi notado. Neste caso, no entanto, o utilizador ainda tem algumas hipóteses de escolha quanto à forma dos assuntos a serem armazenados e, se parecer que um assunto vai ser muito maior do que os outros, pode ser encurtado, regra geral, de preferência a alargar a disposição planeada para arranjar espaço. Se o argumento de economia de espaço não for crucial, as disposições bidimensionais têm a vantagem de possibilitar um espaço claramente identificável para cada assunto, dentro da ficha, o que torna os dados mais fáceis de encontrar.

Permanece, contudo, o problema de que uma tal disposição possa ser inadequada quando se tratar de apagar assuntos ou de os inserir no meio, uma vez que aqueles que se lhe seguirem devem ser movidos um espaço, para cobrir a folga resultante, ou devem fazer espaço para o novo acesso. Visto que não há nenhum comando para mover todo o bloco dentro de uma disposição, isto terá de ser feito assunto por assunto, num processo consumidor de tempo.

Podemos ultrapassar o problema, no que diz respeito à inserção de assuntos, pela simples não inserção de novos assuntos no meio da disposição, no seu lugar correcto, inserindo-os antes no primeiro espaço disponível e deixando a cargo de uma disposição para indicadores a ordem pela qual eles devem ser pegados.

O apagamento não é tão simples. O programa não terá grande utilidade se não se puder retirar assuntos errados. Acresce que, seja qual for a forma como se encare a questão, se o primeiro assunto da disposição for retirado, deixará uma linha em aberto. Em breve, caso precise de apagar assuntos com regularidade, acabará por ter uma disposição crivada de espaços vazios, sem lugar no final para colocar novos registos de dados.

A solução aqui adoptada é a de manter um registo da posição de quaisquer assuntos apagados da ficha, utilizando esse registo como um marcador do lugar onde novos acessos podem ser colocados. Por outras palavras, se o último assunto a ser apagado estivesse na posição um, o assunto a ser introduzido a seguir iria para a posição um. Se, como frequentemente será o caso, não restarem buracos na disposição, o programa colocará o novo assunto no fim dos assuntos correntes. É simples, quando se apanha a ideia, e, como verá logo que introduzir o programa, não se revela difícil na prática.

MÓDULO 4.1.1

```
1000 REM ************************
1010 REM MENU
1020 REM ************************
1030 BORDER 1: INK 0: PAPER 7: C
LS : PRINT "          MultiQ"
1040 PRINT '"COMMANDS AVAILABLE:
"
1050 PRINT '"1)INITIALISE"
1060 PRINT '"2)INPUT NEW ITEMS"
1070 PRINT '"3)SEARCH/DELETE"
```

```
1080 PRINT '"4)ENTER NEW TYPES"
1090 PRINT '"5)GENERATE QUESTION
S"
1100 PRINT '"6)DISPLAY OR RESET
SCORE"
1110 PRINT '"7)STOP"
1120 INPUT "Which do you require
?";Z$
1140 CLS
1150 IF Z$="1" THEN GO SUB 1270
1160 IF Z$="2" THEN GO SUB 1800
1170 IF Z$="3" THEN GO SUB 2460
1180 IF Z$="4" THEN GO SUB 1670
1190 IF Z$="5" THEN GO SUB 2880
1200 IF Z$="6" THEN GO SUB 3380
1210 IF Z$="7" THEN GO TO 1240
1220 CLS
1230 GO TO 1000
1240 PRINT AT 7,12;"MultiQ"
1250 INPUT "Have you entered any
 new          information which yo
u wish to    save? (Y/N)";Q$
1260 IF Q$="Y" THEN SAVE "MULTIQ
": BEEP 1,40: PRINT AT 12,0;"REW
IND, THEN ""ENTER"" TO VERIFY.":
 PAUSE 0: VERIFY "MULTIQ": PRINT
 '"VERIFIED"
1265 STOP
```

Trata-se de um módulo de ementa padrão.

MÓDULO 4.1.2

```
1270 REM ************************
1280 REM INITIALISE
1290 REM ************************
1295 DIM O$(32)
1300 DIM C$(2,20)
1310 LET I$=CHR$ 0+CHR$ 1+CHR$ 0
+CHR$ 2
1320 DIM C(2)
```



```
1330 LET L$="_____________________
_____"
1340 LET ITEMS=2
1350 LET P$="     "
1360 LET ENDITEM=3
1370 LET RIGHT=0
1380 LET QTOT=0
1390 INK 1: PRINT "              TEA
CHER"
1400 PRINT '"NAME FOR ANSWER:";
1410 INPUT C$(1)
1420 PRINT C$(1)
1430 PRINT '"LENGTH OF LONGEST A
NSWER:";
1440 INPUT C(1)
1450 LET C(1)=C(1)+1
1460 PRINT C(1)-1
1470 PRINT ''"NAME FOR QUESTION:
";
1480 INPUT C$(2)
1490 PRINT C$(2)
1500 PRINT '"LENGTH OF LONGEST Q
UESTION:";
1510 INPUT C(2)
1520 PRINT C(2)
1530 INPUT "Are these correct? (
Y/N)";Q$
1550 CLS
1560 IF Q$<>"Y" THEN GO TO 1270
1570 DIM F$(C(1))
1580 DIM G$(C(2))
1590 LET C1=C(1)+C(2)
1600 LET C2=INT (25000/C1)
1610 DIM A$(C2,C(1))
1620 DIM B$(C2,C(2))
1625 LET A$(1,1)=CHR$ 0
1630 LET A$(2,1)=CHR$ 255
1640 LET TYPES=0
1650 DIM D$(10,20)
1660 DIM D(2,10)
```

```
1670 PRINT "INPUT ";"EXTRA " AND
 (TYPES>0);"TYPES OF ANSWER:"
1680 PRINT '"("ZZZ" TO QUIT)"
1720 INPUT Q$
1730 IF Q$="ZZZ" THEN RETURN
1740 LET D$(TYPES+1)=Q$
1750 PRINT TYPES+1;") ";D$(TYPES
+1)
1755 LET TYPES=TYPES+1
1760 IF TYPES<10 THEN GO TO 1720
1770 PRINT '"ONLY 10 TYPES ALLOW
ED."
1780 PRINT '"Press any key to co
ntinue."
1785 PAUSE 0
1790 RETURN
```

Este módulo organiza variáveis e redimensiona as disposições por forma a servir novas aplicações do programa.

Comentário

Linhas 1390-1580. São armazenados dados no programa sob a denominação de perguntas, respostas e categorias, definindo o utilizador os nomes dados a cada uma. Se os testes tomarem a forma de uma palavra inglesa acompanhada por cinco possíveis traduções em francês, poderá inserir «francês» em resposta a «nome para resposta», sendo «inglês» a resposta a «nome para pergunta». Deverá igualmente decidir-se quanto ao comprimento máximo para perguntas e respostas, nesta aplicação particular.

Linhas 1590-1620. As duas disposições principais em que os dados serão armazenados (A$ = respostas, B$ = perguntas) são redimensionadas para poderem receber todo o espaço de memória disponível, sem olhar ao comprimento das linhas em cada uma. Quanto mais curtos forem os assuntos, mais linhas haverá.

Linhas 1650-1790. Pode ser dada uma categoria a cada assunto, ou seja, substantivo, verbo, etc., no caso de um teste de francês. Estas categorias podem ser utilizadas para, mais tarde, tornar os testes mais rigorosos. Podem especificar-se até dez categorias, sendo os respectivos nomes armazenados em D$. A disposição D será utilizada para registar quantos assuntos de cada categoria existem e onde se encontram agrupados na ficha. Note-se que é possível omitir a provisão para categorias, inserindo ZZZ antes da introdução de quaisquer nomes de categorias. Neste caso, o programa não fará referência futura a categorias.

Ensaio do módulo 4.1.2

Quando passado, este módulo deverá pedir denominações e categorias.

MÓDULO 4.1.3

```
1800 REM ************************
1810 REM INPUT OF NEW ITEMS
1820 REM ************************
1830 PRINT "           NEW ITEMS"
1840 PRINT PAPER 6;'C$(1)
1850 INPUT F$
1860 IF ITEMS<>C2 THEN GO TO 189
0
1870 PRINT '"NO ROOM FOR MORE IT
EMS."
1880 GO TO 2130
1890 IF F$(1 TO 3)="ZZZ" THEN GO
 TO 2130
1900 PRINT INVERSE 1;F$
1910 PRINT PAPER 6;'C$(2)
1920 INPUT G$
1930 PRINT INVERSE 1;G$
1940 IF D$(1,1)<>" " THEN GO TO
1970
1950 LET T=0
1960 GO TO 2040
1970 PRINT '"TYPES:"
```

```
1980 FOR I=1 TO TYPES
1990 PRINT I;") ";D$(I)
2000 NEXT I
2010 INPUT T
2030 PRINT '"TYPE SELECTED:";D$(
T)
2040 INPUT "Are these correct? (
Y/N)";Q$
2060 CLS
2070 IF Q$<>"Y" THEN GO TO 1830
2075 IF T<>0 THEN LET D(1,T)=D(1
,T)+1
2080 LET F$=CHR$ T+F$
2090 GO SUB 2190
2100 GO SUB 2330
2110 LET ITEMS=ITEMS+1
2120 GO TO 1800
2130 LET TOTAL=2
2140 FOR I=1 TO 10
2150 LET D(2,I)=TOTAL
2160 LET TOTAL=TOTAL+D(1,I)
2170 NEXT I
2180 RETURN
```

Este módulo aceita novas entradas, sob as denominações e categorias especificadas no módulo anterior.

Comentário

Linha 1860. C2 é o número de linhas na disposição principal.

Linha 1940. Pede-se apenas ao utilizador que especifique uma categoria, caso haja, pelo menos, um nome de uma categoria principal armazenado em D$.

Linha 2080. A categoria —0, caso não haja categorias especificadas— é acrescentada ao inicio da resposta, sob a forma de um marcador de carácter único.

Linha 2130-2170. Este desvio transfere um total acumulativo das somas da primeira coluna na disposição D, ou seja, os números de assuntos de cada categoria, para a segunda coluna da mesma disposição. Uma vez que os assuntos são armazenados por ordem de categoria, isto armazena, efectivamente, o endereço inicial de cada grupo de categorias na segunda coluna.

Ensaio do módulo 4.1.3

O programa deverá aceitar a entrada de um assunto, embora não possa ser colocado na disposição.

MÓDULO 4.1.4

```
2190 REM ************************
2200 REM BINARY SEARCH
2210 REM ************************
2215 DEF FN A()=256*CODE I$(2*S-
1)+CODE I$(2*S)
2220 LET POWER=INT (LN ITEMS/LN
2)
2230 LET S=2^POWER
2240 FOR I=POWER-1 TO 0 STEP -1
2250 LET P=FN A()
2260 LET S=S+(2^I)*(A$(P)<F$)-(2
^I)*(A$(P)>F$)
2270 IF S<2 THEN LET S=2
2280 IF S>ITEMS-1 THEN LET S=ITE
MS-1
2290 NEXT I
2300 LET P=FN A()
2310 IF A$(P)>F$ THEN LET S=S-1
2320 RETURN
```

Este módulo determina a posição correcta de novos assuntos na ordem existente e é paralelo ao módulo de busca, no «Uni-ficha». O método usado é, novamente, o da busca binária.

MÓDULO 4.1.5

```
2330 REM **********************
2340 REM INSERTION
2350 REM **********************
2360 IF LEN P$=4 THEN GO TO 2400
2370 LET PLACE=256*CODE P$(3)+CO
DE P$(4)
2380 LET P$=P$( TO 2)+P$(5 TO )
2390 GO TO 2420
2400 LET PLACE=ENDITEM
2410 LET ENDITEM=ENDITEM+1
2420 LET A$(PLACE)=F$
2430 LET B$(PLACE)=G$
2440 LET I$=I$( TO 2*S)+CHR$ INT
 (PLACE/256)+CHR$ INT (PLACE-256
*INT (PLACE/256))+I$(2*S+1 TO )
2450 RETURN
```

O novo assunto é inserido no primeiro espaço disponivel e o respectivo endereço armazenado na disposição I$.

Comentário

Linhas 2360-2390. P$ armazena os endereços de buracos na disposição. Quando tais buracos são inexistentes, o comprimento da sequência é 4 — o mínimo, de forma a evitar que a linha 2380 apareça com uma mensagem de erro. Desde que P$ seja mais comprida do que quatro caracteres, o segundo par é transferido para um endereço.

Linhas 2400-2410. Se P$ não indicar qualquer folga, o novo assunto será colocado no fim da ficha, no primeiro espaço vazio.

Linhas 2420-2440. A resposta e a pergunta, F$ e G$, são colocadas na linha PLACE das disposições A$ e B$, respectivamente. Note-se que PLACE regista o endereço corrente do assunto da ficha e não a sua posição correcta na ordem, a qual fica guardada na variável S. Isto é determinado pela busca binária. Visto que a categoria foi acrescentada ao principio da resposta, a ordem baseia-se, em primeiro lugar, na categoria.

Ensaio do módulo 4.1.5

Introduza uma série de assuntos da mesma categoria: devem ser introduzidos por ordem alfabética inversa, ou seja, *e, d, c, b, a*. Os assuntos devem ser colocados nas disposições pela ordem em que são introduzidos. Examine agora I$. Deve poder transformar pares de caracteres em indicadores das posições 5, 4, 3, 2, 1, ou de quantos assuntos introduzir em ordem alfabética inversa. Lembre-se de que os assuntos são escolhidos com base nas respostas.

MÓDULO 4.1.6

```
2460 REM **********************
2470 REM USER SEARCH
2480 REM **********************
2485 IF ITEMS=2 THEN RETURN
2490 PRINT "          SEARCH"
2495 PRINT AT 21,15;"TOTAL ITEMS
:";ITEMS-2
2500 PRINT AT 3,0;"COMMANDS AVAI
LABLE:"
2510 PRINT '">""ENTER"" FOR NEXT
 ITEM"
2520 PRINT ">POSITIVE OR NEGATIV
E NUMBER TO MOVE POINTER"
2530 PRINT ">""DDD"" TO DELETE I
TEM"
```

```
2540 PRINT ">""ZZZ"" TO QUIT FUN
CTION"
2550 PRINT 'L$
2580 LET S=2
2600 LET P=FN A( )
2610 PRINT AT 12,0;"ENTRY No. ";
S-1
2620 PRINT 'A$(P,2 TO )
2630 PRINT B$(P)
2640 IF CODE(A$(P,1))=0 THEN GO
TO 2660
2650 PRINT 'D$(CODE A$(P,1))
2660 INPUT S$
2670 IF S$="DDD" THEN GO SUB 278
0: RETURN
2690 IF S$="ZZZ" THEN RETURN
2700 IF S$<>"" THEN GO TO 2740
2710 LET S=S+1
2720 IF S=ITEMS THEN RETURN
2730 GO TO 2600
2740 LET S=S+VAL S$
2750 IF S>=ITEMS THEN RETURN
2760 IF S<2 THEN LET S=2
2770 GO TO 2600
```

O objectivo deste módulo é permitir ao utilizador passar em revista os acessos e apagar os que não forem desejados. Este módulo é mais simples do que o módulo de busca no programa «Unificha», tendo a capacidade de mostrar o acesso seguinte ou de movimentar um número especificado de assuntos. Na realidade, não faz busca.


```
2780 REM ************************
2790 REM DELETIONS
2800 REM ************************
2810 LET P=FN A( )
2815 LET D(1,CODE A$(P,1))=D(1,C
ODE A$(P,1))-1
2820 LET A$(P)=""
2830 LET B$(P)=""
2840 LET I$=I$( TO 2*S-2)+I$(2*S
+1 TO )
2850 LET ITEMS=ITEMS-1
2860 LET P$=P$( TO 2)+CHR$ INT (
P/256)+CHR$ INT (P-256*INT (P/25
6))+P$(3 TO )
2865 GO SUB 2130
2870 RETURN
```

Este módulo retira assuntos da ficha e armazena o endereço da folga resultante em **P$**.

Comentário

Linhas 2810-2860. Uma vez que a variável S marca o assunto a ser mostrado, é usada para extrair um endereço da disposição para indicadores, **I$**, e as linhas deste endereço, em **A$** e **B$**, são esvaziadas. O endereço é removido de **I$** e colocado em **P$**.

Linha 2865. Os endereços iniciais dos vários grupos de categorias são reajustados.

Ensaio do módulo 4.1.7

Apague um assunto e verifique se os restos dos dados continua numa ordem razoável.

```
2880 REM **********************
2890 REM RANDOM QUESTIONS
2900 REM **********************
2910 LET QUESTION=0
2920 PRINT "           QUESTIONS"
2930 PRINT '"DO YOU WISH POSSIBL
E ANSWERS TO BE DRAWN ONLY FROM
THE SAME     ANSWER TYPE? (Y/N)"
2940 INPUT Q$: CLS
2960 IF Q$="Y" THEN LET QUESTION
=1
2970 DIM Q(5)
2980 LET Q1=INT (RND*(ITEMS-2)+2
)
2990 LET S=Q1
3000 LET P=FN A()
3010 LET Q2=INT (RND*5+1)
3020 LET Q(Q2)=P
3030 LET START=2
3040 LET NUMBER=ITEMS-2
3045 IF CODE(A$(P,1))=0 THEN GO
TO 3080
3050 IF QUESTION=0 OR D(1,CODE A
$(P,1))<5 THEN  GO TO 3080
3060 LET START=D(2,CODE A$(P,1))
3070 LET NUMBER=D(1,CODE A$(P,1)
)
3080 FOR I=1 TO 5
3090 IF I=Q2 THEN GO TO 3170
3100 LET S=INT (RND*NUMBER+START
)
3110 LET P=FN A()
3120 IF P=Q(Q2) THEN GO TO 3100
3130 FOR J=1 TO I
3140 IF P=Q(J) THEN GO TO 3100
3150 NEXT J
3160 LET Q(I)=P
```



```
3170 NEXT I
3180 PRINT AT 1,0;C$(2)
3190 PRINT 'B$(Q(Q2))
3200 PRINT 'L$''
3205 PRINT C$(1)''
3210 FOR I=1 TO 5
3220 PRINT I;") ";A$(Q(I),2 TO )
3230 NEXT I
3240 PRINT ''"WHICH DO YOU THINK
 IS THE RIGHT ANSWER?"'"TYPE IN
THE NUMBER:";
3250 LET QTOT=QTOT+1
3260 INPUT ANSWER: PRINT "";ANSW
ER
3270 IF ANSWER=Q2 THEN GO TO 330
0
3280 PRINT '"INCORRECT. THE RIGH
T ANSWER WAS:";A$(Q(Q2),2 TO )
3290 GO TO 3320
3300 FOR I=14 TO 20
3302 PRINT AT I,0;O$
3304 NEXT I
3306 FLASH 1: PRINT AT 8+ANSWER,
3;A$(Q(ANSWER),2 TO ): PRINT AT
16,10;"CORRECT": PAUSE 50: FLASH
 0
3310 LET RIGHT=RIGHT+1
3320 PRINT """ENTER"" for new qu
estion or     ""ZZZ"" to quit fu
nction."
3340 INPUT Q$
3350 CLS
3360 IF Q$="ZZZ" THEN RETURN
3370 GO TO 2970
```

Este módulo organiza os múltiplos testes de respostas, com base nos assuntos armazenados.

Comentário

Linhas 2880-3000. O utilizador tem a opção de escolher entre tirar as cinco respostas de cada pergunta de toda a ficha, ou apenas de assuntos da mesma categoria.

Linha 2980-3000. É escolhido um assunto da ficha, ao acaso.

Linhas 3010-3020. Os endereços das cinco respostas serão armazenados na disposição Q. O endereço da resposta correcta é colocado numa posição casual, dentro da disposição.

Linhas 3030-3170. Ao criar um número casual, dentro de certos limites, é necessário saber duas coisas: em primeiro lugar, a base em que o número irá assentar e, em segundo lugar, os limites acima dessa base em que o número deverá cair. A função RND do *Spectrum* produz um valor pseudocasual entre 0 e 1, pelo que um número com a forma de X + INT (Y*RND) será capaz de variar entre X e X + Y—1. O valor mínimo que Y*RND pode comportar poderá ser inferior a 1, pelo que INT(Y*RND) será igual a 0. O valor máximo de INT(Y*RND) é Y—1, uma vez que RND nunca poderá realmente igualar 1.

Nesta secção do programa deve ser gerado um número de endereços ao acaso, sendo isto feito na base das variáveis START e NUMBER. Se o utilizador especificou que as respostas serão tiradas de toda a ficha, em vez de o serem de uma categoria especifica, START é colocada em 2, que é o primeiro assunto útil após o princípio do indicador da ficha. NUMBER é regulada para o verdadeiro número de assuntos — ITEMS menos os dois indicadores da ficha.

Se o utilizador apenas especificou respostas da mesma categoria, START é regulada para o primeiro endereço daquele grupo sobre que recaiu a escolha casual e NUMBER é regulada para o número de assuntos dentro do grupo. Isto apenas será feito se houver cinco ou mais assuntos no grupo em questão. A resposta acabada de chegar é comparada com a resposta original, para se ver que não é a mesma, e comparada em seguida com o resto das respostas escolhidas ao acaso. Se a nova escolha casual não for uma duplicação, o respectivo endereço é colocado em Q, numa posição casual.

Linhas 3180-3230. A pergunta é impressa no topo do *écran;* as respostas são impressas ao fundo.

Ensaio do módulo 4.1.8

Introduza algumas perguntas e respostas e verifique você próprio.

MÓDULO 4.1.9

```
3380 REM ************************
3390 REM SCORE
3400 REM ************************
3410 PRINT "              SCORE"
3420 IF QTOT<>0 THEN GO TO 3470
3430 PRINT '''''"       NO SCORE Y
ET."
3440 PRINT '"   Any key to conti
nue."
3450 PAUSE 0
3460 RETURN
3470 PRINT '"TOTAL QUESTIONS:";Q
TOT
3480 PRINT '"CORRECT:";RIGHT
3490 PRINT "ASSESSMENT:";INT ((
RIGHT-QTOT/5)/(QTOT*.8))*100);"
PER CENT."
3500 PRINT '"DO YOU WISH TO ZERO
 SCORE? (Y/N)"
3510 INPUT Q$
3520 IF Q$<>"Y" THEN RETURN
3530 LET QTOT=0
3540 LET RIGHT=0
3550 RETURN
```

Este módulo está concebido para mostrar os resultados correntes, anulando-os para recomeçar, caso seja pedido.

Comentário

Linha 3490. Esta linha é uma tentativa de avaliação razoável dos resultados, dado que carregar em botões ao acaso resultará numa pontuação de 20%, em média. Estes hipotéticos 20% são subtraídos, tanto do total de perguntas como das respostas correctas, sendo a avaliação feita sobre o resto.

Ensaio do módulo 4.1.9

Se ensaiou anteriormente o módulo 8, as variáveis já foram fixadas. A solicitação deste módulo fornecer-lhe-á uma pontuação e uma avaliação.

Sumário

Trata-se, na verdade, de um programa bastante poderoso, o que apenas poderá confirmar por si próprio se lhe fornecer algum material. Muito à parte da sua utilidade como ferramenta educacional, o programa é também uma simples demonstração de um método de aceleração do acesso a disposições, através do afastamento da necessidade de esquadrinhar a ficha de cada vez que se faz um apagamento. O núcleo do programa poderia facilmente ser aplicado a uma vasta gama de utilizações, onde a velocidade do acesso aos assuntos é mais importante do que poupar os últimos *bytes* do espaço de memória.

Avançando mais

1) O programa, tal como se apresenta, não dá grandes recompensas para respostas certas ou boas pontuações. Poderá introduzir-lhe qualquer tipo de recompensa visual, num ponto apropriado?

2) A programação é uma prática bastante solitária mas, utilizando um programa destes, pode-se melhorar através da colaboração com outros. Convença outrem a comprar este livro e ponha-o a construir uma ficha sobre Física, enquanto você acrescenta alguma História.

3) Poderá adicionar uma função de busca mais poderosa, baseada na função de busca do «Unificha»?

4.2 PALAVRAS

A moral deste programa reside em que, quando se tem um método que funciona bem, deve-se adaptá-lo para realizar uma variedade de tarefas, ainda que o resultado seja suspeitamente semelhante a outros programas escritos anteriormente por si. Este programa é uma imitação de «MultiQ», residindo a única diferença real na apresentação das programas, que tomam a forma de imagens. Foi feito para ser um explicador de leitura mas, é claro, é igualmente aplicável a qualquer outro tópico onde tudo deva ser apresentado visualmente e feitas perguntas.

A capacidade do programa é bastante limitada, uma vez que a pequena imagem tem reservado para ela um espaço de 400 caracteres. As próprias imagens são as geradas pelo programa «Artista». Isto significa que a sequência de 400 caracteres representa uma versão compacta de algo que, originalmente, tinha um máximo de 120 caracteres, aproximadamente. O desenho deve utilizar apenas a metade inferior do *écran*.

MÓDULO 4.2.1

```
1000 REM ************************
1010 REM MENU
1020 REM ************************
1030 BORDER 1: INK 0: PAPER 6: C
LS
1040 PRINT "           WORDS"
1050 PRINT '"COMMANDS AVAILABLE
"
1060 PRINT '"1)INITIALISE"
1070 PRINT '"2)INPUT NEW ITEMS"
1080 PRINT '"3)SEARCH/DELETE"
1090 PRINT '"5)GENERATE QUESTION
S"
1100 PRINT '"6)DISPLAY OR RESET
SCORE"
1110 PRINT '"7)STOP"
```

```
1120 INPUT "Which do you require
?";Z$
1130 CLS
1140 IF Z$="1" THEN GO SUB 1260
1150 IF Z$="2" THEN GO SUB 1390
1160 IF Z$="3" THEN GO SUB 1680
1170 IF Z$="5" THEN GO SUB 2010
1180 IF Z$="6" THEN GO SUB 2340
1190 IF Z$="7" THEN GO TO 1230
1200 CLS
1210 GO TO 1000
1220 PRINT AT 7,12;"WORDS"
1230 INPUT "Have you entered any
 new          information which yo
u wish to    save? (Y/N)";Q$
1240 IF Q$="Y" THEN SAVE "WORDS"
: PRINT AT 10,0;"REWIND, THEN AN
Y KEY TO VERIFY.": PAUSE 0: VERI
FY "WORDS"
1250 STOP
```

Trata-se de um módulo de ementa padrão.

MÓDULO 4.2.2

```
1260 REM **********************
1270 REM INITIALISE
1280 REM **********************
1290 DIM O$(32)
1300 LET L$="________________________
_____"
1310 LET ITEMS=0
1320 LET I$="     "
1330 LET ENDITEM=1
1340 LET RIGHT=0
1350 LET QTOT=0
1360 DIM A$(50,15): DIM B$(50,40
0): DIM C$(15)
```



```
1370 DIM B(100)
1380 RETURN
```

Este módulo contém as variáveis do programa. As disposições são utilizadas para guardar a sequência compacta e a palavra associada.

MÓDULO 4.2.3

```
2620 REM **********************
2630 REM PRINT PACKED STRING
2640 REM **********************
2650 BORDER 1: PAPER 7: CLS : PR
INT AT 1,1;: LET SCREEN=0: LET C
OUNTER=1
2660 IF P$(COUNTER)=CHR$ 255 THE
N LET COUNTER=COUNTER+1: IF P$(C
OUNTER)<>CHR$ 0 THEN PRINT O$( T
O CODE P$(COUNTER));: LET SCREEN
=SCREEN+CODE P$(COUNTER): PRINT
" " AND SCREEN/30=INT (SCREEN/3
0);: LET COUNTER=COUNTER+1: IF C
OUNTER<=LEN P$ THEN GO TO 2660
2670 IF COUNTER>LEN P$ THEN GO T
O 2710
2680 IF P$(COUNTER)=CHR$ 0 THEN
LET COUNTER=COUNTER+1
2690>LET SCREEN=SCREEN+1: INVERS
E CODE P$(COUNTER+2): PRINT P$(C
OUNTER);: POKE (23296-32*(PEEK 2
3689)-PEEK 23688+32),CODE P$(COU
NTER+1): INVERSE 0: IF SCREEN/30
```

```
=INT (SCREEN/30) THEN PRINT "   "
;
2700 LET COUNTER=COUNTER+3: IF C
OUNTER<=LEN P$ THEN GO TO 2660
2710 PAPER 7: INK 1: RETURN
```

Este módulo é retirado directamente do programa «Artista» e reimprime uma sequência compacta.

MÓDULO 4.2.4

```
1560 REM ************************
1570 REM INSERTION
1580 REM ************************
1590 IF LEN I$=4 THEN GO TO 1630
1600 LET PLACE=256*CODE I$(3)+CO
DE I$(4)
1610 LET I$=I$( TO 2)+I$(5 TO )
1620 GO TO 1650
1630 LET PLACE=ENDITEM
1640 LET ENDITEM=ENDITEM+1
1650 LET A$(PLACE)=W$
1660 LET B$(PLACE)=P$: LET B(PLA
CE)=LEN P$
1670 RETURN
```

Este tem a mesma estrutura do módulo de inserção, no programa anterior, excepto que a sequência que regista os buracos da disposição se chama I$ e o endereço do novo acesso não é armazenado numa disposição para indicadores. Com apenas 50 acessos, parecia desnecessário ter uma função de busca complexa para estabelecer uma ordem correcta de acessos.

MÓDULO 4.2.5

```
1390 REM ************************
1400 REM INPUT OF NEW ITEMS
1410 REM ************************
1420 PRINT "           NEW ITEMS"
1430 INPUT "Name of Picture to b
e loaded     from tape? (ZZZ TO Q
UIT)";N$
1440 PRINT '''"PRESS ANY KEY THEN
 START TAPE.": PAUSE 0: PRINT ''
"          LOADING NOW": LOAD N$
 DATA P$( )
1450 GO SUB 2620
1460 INPUT "Do you want this Pic
ture?(Y/N)";Q$
1470 IF Q$<>"Y" THEN RETURN
1480 INPUT "Word to go with this
 Picture?";W$
1490 PRINT AT 21,0;W$
1500 INPUT "Is this correct? (Y/
N)";Q$
1510 CLS : IF Q$="N" THEN GO TO
1480
1520 GO SUB 1560: LET ITEMS=ITEM
S+1
1530 INPUT "Do you want to inPut
 another    Picture? (Y/N)":Q$
1540 IF Q$="N" THEN RETURN
1550 GO TO 1420
```

Este módulo retira da fita magnética a sequência compacta que contém o desenho, imprime-o, aceita a palavra que lhe corresponde e, finalmente, armazena-o.

Ensaio do módulo 4.2.5

Após ter introduzido vários módulos de aspecto familiar, sem ensaiar o seu funcionamento, você está agora em posição de verificar se o programa retira disposições da fita — disposições que foram criadas e salvas utilizando o programa «Artista» —, passa no *écran* os desenhos que elas contêm e os armazena na memória.

MÓDULO 4.2.6

```
1680 REM ************************
1690 REM USER SEARCH
1700 REM ************************
1710 IF ITEMS=0 THEN RETURN
1720 LET S=1
1730 IF B$(S,1)=" " AND S<50 THE
N LET S=S+1: GO TO 1730
1740 IF S=50 AND B$(S,1)=" " THE
N RETURN
1750 CLS : DIM P$(B(S)): LET P$=
B$(S,1 TO B(S)): GO SUB 2620: PR
INT AT 21,0;A$(S)
1760 PRINT AT 0,0;"            S
EARCH"
1770 PRINT '"COMMANDS AVAILABLE:
"
1780 PRINT '">""ENTER"" FOR NEXT
 ITEM"
1790 PRINT ">POSITIVE OR NEGATIV
E NUMBER TO MOVE POINTER"
1800 PRINT ">""DDD"" TO DELETE I
TEM"
1810 PRINT ">""ZZZ"" TO QUIT FUN
CTION"
1820 PRINT 'L$
```



```
1830 INPUT S$
1840 IF S$="DDD" THEN GO SUB 193
0: RETURN
1850 IF S$="ZZZ" THEN RETURN
1860 IF S$<>"" THEN GO TO 1890
1870 LET S=S+1
1880 GO TO 1730
1890 LET S=S+VAL S$
1900 IF S>=50 THEN RETURN
1910 IF S<1 THEN LET S=1
1920 GO TO 1730
```

Esta é, novamente, uma versão simplificada do módulo de busca retirado do programa anterior. Ao mostrar um acesso em particular, o módulo não pede, em primeiro lugar, o endereço a uma disposição para indicadores, começando simplesmente na primeira linha da disposição principal e imprimindo o que quer que lá se encontre, a menos que a posição do primeiro carácter seja um espaço. Se for o caso, a linha estará vazia e o módulo passará à linha seguinte.

Ensaio do módulo 4.2.6

Deverá poder passar em revista os acessos armazenados, quer para a frente, quer para trás.

MÓDULO 4.2.7

```
1930 REM ************************
1940 REM DELETIONS
1950 REM ************************
1960 LET A$(S)=""
1970 LET B$(S)=""
1980 LET ITEMS=ITEMS-1
1990 LET I$=I$( TO 2)+CHR$ INT (
S/256)+CHR$ INT (S-256*INT (S/25
6))+I$(3 TO )
2000 RETURN
```

Este módulo permite ao utilizador apagar acessos, e é uma versão simplificada do mesmo módulo, no programa anterior.

Ensaio do módulo 4.2.7

Deverá agora poder apagar acessos à vontade.

MÓDULO 4.2.8

```
2010 REM ************************
2020 REM RANDOM QUESTIONS
2030 REM ************************
2040 LET QUESTION=0
2050 DIM Q(5)
2060 LET Q1=INT (RND*ITEMS+1)
2070 LET Q2=INT (RND*5+1)
2080 LET Q(Q2)=Q1
2090 FOR I=1 TO 5
2100 IF I=Q2 THEN GO TO 2170
2110 LET S=INT (RND*ITEMS+1)
2120 IF S=Q(Q2) THEN GO TO 2110
2130 FOR J=1 TO I
2140 IF S=Q(J) THEN GO TO 2110
2150 NEXT J
2160 LET Q(I)=S
2170 NEXT I
2180 DIM P$(B(Q1)): LET P$=B$(Q1
): GO SUB 2620
2190 PRINT AT 0,0;
2200 FOR I=1 TO 5: DIM T$(32): L
ET T$=A$(Q(I))
2210 PRINT INK 9; PAPER 7-I;T$
2220 NEXT I
2230 PRINT '"WHICH ONE?"
2240 LET QTOT=QTOT+1
```



```
2250 INPUT C$: PRINT C$: IF C$=A
$(Q1) THEN GO TO 2270
2260 PRINT '"WRONG-";A$(Q1): GO
TO 2300
2270 FOR I=0 TO 10: LET I1=I-8*(
I>7): PRINT AT I,0; OVER 1: PAPE
R I1;Q$: NEXT I
2280 FLASH 1: PRINT AT 02-1,13;A
$(Q1): PRINT AT 10,10;"RIGHT!":
PAUSE 200: FLASH 0
2290 LET RIGHT=RIGHT+1
2300 INPUT """ENTER"" for new qu
estion or      ""ZZZ"" to stop";Q
$
2310 CLS
2320 IF Q$="ZZZ" OR Q$="zzz" TH
EN GO TO 2520
2330 GO TO 2050
```

Este módulo fabrica as perguntas casuais e possíveis respostas. É menos complexo do que o módulo equivalente no programa anterior, pois não há qualquer provisão para categorias de acessos.

Comentário

Linha 2200-2280. Note-se como é possível tornar visualmente atraente a apresentação das perguntas e da recompensa por uma resposta certa, sem grandes esforços. Nestas linhas são utilizadas variáveis de desvio para estabelecer cores de fundo, tornando o *écran* mais interessante.

Ensaio do módulo 4.2.8

O programa deverá agora ser capaz de gerar perguntas.

```
2340 REM ************************
2350 REM SCORE
2360 REM ************************
2370 PRINT "                SCORE"
2380 IF QTOT<>0 THEN GO TO 2430
2390 PRINT '''''"       NO SCORE Y
ET."
2400 PRINT '"   Any key to conti
nue."
2410 PAUSE 0
2420 RETURN
2430 PRINT '"TOTAL QUESTIONS:";Q
TOT
2440 PRINT '"CORRECT:";RIGHT
2450 DEF FN A()=INT (((RIGHT-QTO
T/5)/(QTOT*.9))*100): PRINT '"AS
SESSMENT:";FN A();" PER CENT."
2460 PRINT '"DO YOU WISH TO ZERO
 SCORE? (Y/N)"
2470 INPUT Q$
2480 IF Q$<>"Y" THEN RETURN
2490 LET QTOT=0
2500 LET RIGHT=0
2510 RETURN
```

Este é o módulo de pontuação, tirado do programa anterior, com uma pequena alteração.

Comentário

Linha 2450. A avaliação, em vez de ser impressa directamente no *écran*, é armazenada na variável ASS para posterior utilização.

Ensaio do módulo 4.2.9

Deverá agora poder solicitar à ementa uma pontuação.


```
2520 REM ************************
2530 REM BYE
2540 REM ************************
2550 CLS : LET I1=0: LET I2=1: P
APER 9
2560 FOR I=1 TO 100
2570 LET I1=I1+1-25*(I1=24)
2580 LET I2=I2+1-8*(I2=7)
2590 POKE 23692,255: INK I2: PRI
NT 'TAB I1;"GOODBYE!";
2600 NEXT I: PRINT " YOU SCORED
";FN A()
2610 INK 0: PAPER 7: STOP
```

Este módulo imprime no *écran* uma despedida menos enfadonha ao programa e encarrega a função definida no módulo de pontuação de fazer uma avaliação final.

Comentário

Linha 2590. Note-se a utilização deste empurrão para evitar que a função *scroll* (inventário) pare, após terem sido impressas 22 linhas. O endereço 23692 da memória é utilizado pelo *Spectrum* para registar o número de linhas que permanecem por imprimir, antes de ser perguntado ao utilizador se o inventário deve prosseguir.

Ensaio do módulo 4.2.10

Este módulo apenas pode ser solicitado do módulo que gera as perguntas.

Sumário

Este é, de novo, um programa que requer algum trabalho para ter alguma utilidade, uma vez que os pequenos desenhos que utiliza levam algum tempo a criar. Ao criar os desenhos, é especialmente importante prepará-los correctamente, antes de os gravar com o programa «Artista». Se utilizar papel quadriculado, poderá contar o número de caracteres do desenho. Tudo o que passe dos 100, ou cujos caracteres se encontrem demasiado espalhados pelo *écran*, não deverá caber no espaço de 400 caracteres reservado para a sequência compacta. Já terá notado, provavelmente, que a parte do programa que administra o teste está bastante desligada das outras funções. Isto está feito assim para que o adulto possa pôr o teste a funcionar e deixá-lo, sabendo que será pouco provável que a criança reintroduza o programa principal, fazendo estragos ou batota. Antes de iniciar o teste, talvez seja melhor passar CAPS LOCK a tipo minúsculo, visto que as crianças tendem a familiarizar-se mais com esse formato de letra. Para que isto funcione, as palavras associadas a cada imagem deverão igualmente ter sido introduzidas em tipo minúsculo. Se ainda não tiver crianças com a idade apropriada, poderá sempre desenhar, para si próprio, um conjunto de caracteres representando componentes eléctricos, preparando o programa para lhe ensinar a identificar circuitos electrónicos simples. Tal como já disse antes, se possui algo que funciona, adapte-o.

Avançando mais

1) Se realmente não precisa de um registo da característica inversa de cada carácter da sequência compacta, tanto pode aumentar o número de desenhos que o programa pode armazenar como aumentar a sua complexidade.

2) Aqui está outro caso em que a colaboração com outros possuidores do *Spectrum*, na permuta de desenhos, lhe pode poupar bastantes esforços.

4.3 ONDE?

Trata-se de um programa pouco complicado, que ensaia, efectivamente, os seus conhecimentos de geografia, ou, pelo menos, da localização de cidades num conjunto de países. Para utilizar o programa, terá de modificar o programa «Artista», dado acima. A modificação consiste no acrescento de um módulo que retoma a grelha de 20*30, contida em A$(1), salvando-a para fita magnética. A realização desta modificação fica ao seu cuidado. A sua escolha poderá ser a de pôr de lado uma versão de «Artista», que apenas cria e funciona com uma disposição A$(20,30) e não armazena características cromáticas ou cria uma sequência compacta, ou a de acrescentar o programa «Artista» já existente, o qual, com um simples desvio do tipo FOR I = 1 TO 20 : LET B$(I) = A$(1,I) : NEXT I, transfere o conteúdo de A$ para outra disposição.

De qualquer das formas, deverá resultar um programa que seja capaz de salvar uma sequência de 20*30 contendo um *écran* cheio de caracteres. O passo seguinte será criar alguns desenhos que produzam as fronteiras de um país, armazenando depois cada um sob o nome do país que representa. É então que se apanha o fio à meada deste programa.

MÓDULO 4.3.1

```
1000>REM **********************
1010 REM MENU
1020 REM **********************
1030 PRINT "          WHERE?"
1040 PRINT '"1)LOAD NEW COUNTRY"
1050 PRINT '"2)RECORD NEW CITIES
"
1060 PRINT '"3)SET QUESTIONS"
1070 PRINT '"4)INITIALISE"
1080 PRINT '"5)STOP"
1090 INPUT Z$: CLS : LET QTOTAL=
0: LET WRONG=0
```

```
1100 IF Z$="1" THEN GO SUB 1260
1110 IF Z$="2" THEN GO SUB 1320
1120 IF Z$="3" THEN LET COUNTRY=
FN A(): GO SUB 1670: GO SUB 1510
1130 IF Z$="5" THEN GO TO 1160
1140 IF Z$="4" THEN GO SUB 1190
1150 CLS : GO TO 1000
1160 PRINT AT 10,10;"WHERE?"
1170 INPUT "Have you entered any
 new data  that you wish to rec
ord? (Y/N)";Q$: IF Q$="Y" THEN S
AVE "WHERE?": PRINT "Rewind then
 ""ENTER"" to verify.": PAUSE 0:
VERIFY "WHERE?"
1190 STOP
```

Trata-se de um módulo de ementa padrão.

Módulo 4.3.2

```
1190 REM ************************
1200 REM INITIALISE
1210 REM ************************
1220 DIM B$(10,20,30)
1230 LET COUNTRIES=0
1240 DEF FN A()=INT (RND*COUNTRI
ES+1)
1250 RETURN
```

São aqui inicializadas as variáveis.

Comentário

Linha 1220. As sequências contendo os países são armazenadas na disposição **B$**.

Linha 1240. Esta função escolhe um país ao acaso.

Módulo 4.3.3

```
1260 REM ************************
1270 REM LOAD NEW COUNTRY
1280 REM ************************
1290 INPUT "NAME FOR COUNTRY TO
BE LOADED? ";N$: INPUT "START TA
PE THEN ""ENTER""";Q$: LOAD N$ D
ATA A$()
1300 INPUT "NUMBER TO BE LOADED
AT:";PLACE
1310 LET COUNTRIES=COUNTRIES+1:
FOR I=1 TO 20: LET B$(PLACE,I)=A
$(I): NEXT I: RETURN
```

Este módulo aceita uma disposição de 20*30 a partir da fita, sob o nome do país relevante, colocando em B$ essa disposição, decidindo o utilizador o que fazer com a posição.

Ensaio do módulo 4.3.3

Introduza uma sequência e coloque-a em B$: verifique em modo directo se ela foi devidamente armazenada.

```
1320 REM ************************
1330 REM RECORD CITIES
1340 REM ************************
1350 INPUT "NUMBER OF COUNTRY? "
;COUNTRY
1360 PRINT : FOR I=1 TO 20: PRIN
T PAPER 5; INK 2;TAB 1;B$(COUNTR
Y,I): NEXT I: LET X=1: LET Y=1:
LET T$="": GO TO 1430
1370 PRINT OVER 1; PAPER 8; INK
0;AT X,Y;"*": PAUSE 2: PRINT OVE
R 1; PAPER 8; INK 0;AT X,Y;"*"
1380 LET T$=INKEY$: IF T$="" THE
N GO TO 1370
1390 IF T$<"5" OR T$>"9" THEN GO
 TO 1370
1400 PRINT PAPER 5; INK 2;AT X,Y
;B$(COUNTRY,X,Y)
1410 LET X=X+(T$="6")-(T$="7"):
LET X=X+(X<1)-(X>20)
1420 LET Y=Y+(T$="8")-(T$="5"):
LET Y=Y+(Y<1)-(Y>30)
1430 PRINT INK 0;AT 0,0;"X=";X;"
  Y=";Y;"  ""9"" TO STOP": IF T$
<>"9" THEN GO TO 1370
1440 PRINT AT 5,0;"A TYPICAL DAT
A ENTRY LOOKS LIKE THIS:"'"5100
DATA 5,""TOYLAND""'"5101 DATA "
"TOYTOWN"",6,12"
1450 PRINT AT 10,0;"RECORD CITY
NAME, X & Y IN DATA STATEMENT AF
TER ";5000+COUNTRY*100;"."
1460 PRINT '"INCREASE NUMBER IN
DATA STATEMENT AT ";5000+COUNTRY
*100;" BY 1."
1470 PRINT '"SECOND ITEM I: THAT
```



```
 LINE SHOULD BE THE COUNTRY NAME
."'"ANY KEY TO LIST RELEVANT DAT
A."
1480 PAUSE 0: LIST 5000+COUNTRY*
100: STOP
```

Este módulo permite ao utilizador movimentar um cursor pelo desenho, até serem estabelecidas as coordenadas de uma cidade a contento do utilizador. Estas coordenadas podem, então, ser inseridas num enunciado de dados no final do programa.

Comentário

Linha 1360. A secção relevante da disposição B$ é impressa no *écran*.

Linhas 1370-1420. Trata-se de rotinas padrão, para fazer movimentar o cursor.

Linha 1430. As coordenadas do cursor são passadas no *écran*.

Linhas 1440-1480. Se o utilizador parar o programa, serão dadas instruções para a inserção das coordenadas num enunciado de dados, no final do programa. A estrutura do enunciado de dados é a seguinte:

1) O nome de cada país é armazenado numa denominação de dados numerada 5000+(100 vezes o seu número em B$). O nome do país é precedido, nessa linha de dados, pelo número das respectivas cidades cujas coordenadas estejam registadas.

2) Os dados relativos a cidades são armazenados na linha imediatamente a seguir, por exemplo, 5101, 5102, sob a forma de linhas únicas, cada uma contendo o nome de uma cidade, seguida pelas coordenadas X e Y.

Se a cidade for a primeira a ser introduzida, para um país acabado de entrar, o país deve ser registado no lugar indicado pelo programa. Se não for a primeira cidade, o assunto que regista o número de cidades deve ser aumentado e a nova cidade inserida numa linha própria. O programa fornece ao utilizador o número das linhas correctas onde introduzir os dados e duas linhas para exem-

plos. Com a utilização, este método de inserção de assuntos revela-se extremamente simples e serve como outro exemplo das economias que podem fazer-se na complexidade de certos programas, desde que o utilizador faça alguma pré-embalagem da informação em que o programa deve trabalhar. Sem a utilização de linhas de dados, teríamos de providenciar uma disposição que contivesse os nomes das cidades e respectivas coordenadas, juntamente com um método de escolha e um método de apagamento.

Ensaio do módulo 4.3.4

Deverá agora poder movimentar o cursor livremente e parar o programa com a inserção de «9», após o que lhe serão dadas instruções para a introdução das coordenadas das cidades no enunciado de dados.

MÓDULO 4.3.5

```
1670 RESTORE (5000+100*COUNTRY):
 READ N,C$: LET CITY=INT (RND*N+
1): RESTORE (5000+100*COUNTRY+CI
TY): READ M$,X,Y: RETURN
```

Esta linha lê os dados referentes a um país e cidade indicados pelas variáveis COUNTRY e CITY. Trata-se de um bom exemplo da flexibilidade da função RESTORE (reconstituição) no *Spectrum*.

MÓDULO 4.3.6

```
1510 REM **********************
1520 REM WHICH CITY
1530 REM **********************
1540 CLS : PRINT INK 0;C$: FOR I
=1 TO 20: PRINT PAPER 5: INK 2;T
AB 1;B$(COUNTRY,I): NEXT I
1550 PRINT OVER 1: PAPER 8; INK
0: FLASH 1;AT X,Y;"*"
1560 FOR I=1 TO 3
1570 LET QTOTAL=QTOTAL+1: INPUT
"NAME OF THIS CITY? ";N$: IF N$=
M$ THEN GO TO 1620
1580 PRINT AT 0,0;"WRONG!
                  ": PAUSE 100:
 PRINT AT 0,0;C$;" ": LET WRONG=
WRONG+1: NEXT I
1590 PRINT AT 0,0;"THE CITY WAS:
";M$: INPUT """ENTER"" TO CONTIN
UE";Q$
1600 PRINT INK 2; PAPER 5; FLASH
 0;AT X,Y;B$(COUNTRY,X,Y)
1610 GO TO 1630
1620 PRINT AT 0,0;"";M$;"
 ": FOR I=1 TO 20: PRINT FLASH 1
; INK 2; PAPER 5;AT I,1;B$(COUNT
RY,I): PRINT INK 1;AT I,1;"RIGHT
!": NEXT I: PRINT OVER 1; FLASH
1;AT X,Y;"*"
1630 PRINT INK 0;AT 21,0;"QUESTI
ONS:";QTOTAL;" RIGHT:";QTOTAL-WR
ONG;" %:";(QTOTAL-WRONG)/QTOTAL*
100
1640 INPUT "ANOTHER GO? (Y/N)? "
;Q$: IF Q$="N" THEN RETURN
```

```
1650 INPUT "SAME COUNTRY? (Y/N)?
 ";Q$: IF Q$="N" THEN LET COUNTR
Y=FN A( )
1660 GO SUB 1670: GO TO 1510
```

Este módulo imprime um país, acrescenta um indicador à posição de uma cidade e pede a inserção do nome de uma cidade. Com base nas respostas dadas até aqui, é impressa uma avaliação da actuação do utilizador até ao momento.

Comentário

Linha 1570. O nome inserido é comparado e verificado em relação ao nome retirado dos dados. Note-se que é possível fazer três tentativas para cada cidade.

Linha 1620. È dada uma simples recompensa visual, caso o utilizador introduza o nome correcto.

Linha 1650. Note-se o uso da função para determinar um novo país, caso o utilizador queira mudar de países.

Ensaio do módulo 4.3.6

O programa deverá agora ser capaz de organizar um teste, baseado nos dados que você introduziu.

Sumário

É possível fazer mau uso dos enunciados de dados. Muitas pessoas o fazem, empregando-os em programas onde frequentemente se torna necessário mudar ou seleccionar dados, de uma ou outra forma. Se colocados no devido lugar, contudo, os enunciados de dados podem ser convenientes e economizar tempo, uma vez que se pode acrescentar ou apagar dados sem a constante reordenação de enormes disposições. Não pode haver uma regra rígida mas,

se um programa estiver sempre a pedir ao utilizador para ser parado e introduzir dados, então é porque o programador está apenas a evitar a tarefa de criar um programa que possa tratar os tipos de dados necessários aos verdadeiros objectivos.

Avançando mais

1) Poderá alargar o programa, de forma a que ele possa imprimir rios ou montanhas, ou outras características físicas, pedindo depois o respectivo nome? Não é fácil, embora seja possível e resulte bastante impressionante.

2) Ocorrem-lhe outras aplicações para este programa? Uma de que me lembro é a de elaborar perguntas sobre a análise de circuitos, para aqueles que estão a aprender electrónica, como seja o valor da corrente no ponto indicado pelo cursor.

CAPÍTULO 5

UTILIDADES — UMA COLECÇÃO DE ROTINAS VARIADAS

Neste capitulo abordamos seis programas muito dispares, sob a imprecisa designação de «utilidades», ilustrando, quanto mais não seja, a variedade de aplicações que podem seleccionar-se para programas dedutíveis, de modo a satisfazer as suas necessidades. Os seis programas são:

1) *Calculadora,* desenhado para tornar o seu *Spectrum* mais útil quando se trata de fazer séries de cálculos repetitivos.
2) *Calorias,* um programa que talvez ilustre as preocupações pessoais do autor, mas que, ainda assim, é um instrumento conveniente para aqueles que precisam de vigiar o peso.
3) *Gráfico,* um programa de aplicação geral, que desenha gráficos.
4) *Renumeração,* uma ferramenta vital para a sua própria programação, permitindo-lhe arrumar programas irregularmente numerados.
5) *Unificha II.* Retomamos o «Unificha» e desenvolvemo-lo de modo a que ele possa aceitar dados de dimensão e estrutura irregular, tornando-se um programa bastante mais poderoso.
6) *Dactilógrafo,* um programa simples, que pode ensinar-lhe a escrever concisamente.

5.1 CALCULADORA

Pode parecer estranho que se faça um programa para tornar mais fáceis os cálculos em computador. Afinal, não é o *Spectrum* capaz de aceitar fórmulas em modo directo para cálculos dedutivos, ou sob a forma de programas directos para cálculos regulares?

A resposta, é claro, é que o *Spectrum* pode fazer ambas as coisas. Mas, entre o cálculo dedutivo que pode facilmente ser introduzido e o uso feito por especialistas, com uma regularidade que justifica a elaboração de um programa especial, tal como acontece com os programas financeiros no cap. 2, encontra-se uma gama de necessidades que requerem a utilização de cálculos repetitivos. Seria demasiado fastidioso introduzir estes últimos em modo directo e, no entanto, eles não assumem uma importância que justifique a elaboração de um programa especial. Para tais aplicações precisamos de um programa que permita a introdução de uma gama de fórmulas e variáveis e respectiva alteração, incluindo a exposição de resultados.

Tendo iniciado um tal projecto, podemos também, simultaneamente, desenhar o programa por forma a que os resultados da alteração do valor de variáveis possa ser comparado seguindo o modelo de um programa «*what if*» (e se...), como o programa orçamental no cap. 2.

Para atingir este objectivo, sem ter de escrever constantemente novas fórmulas em linhas programáticas, faremos uso de duas das características mais atraentes do *Basic* do *Sinclair*: o seu manuseamento flexivel das sequências e capacidade para avaliar uma expressão numérica armazenada numa sequência. Tomadas em conjunto, estas duas características permitir-nos-ão realizar tarefas que seriam perfeitamente impossíveis em muitas outras máquinas cujo preço excede, em muito, o do *Spectrum*.


```
1000 REM ************************
1010 REM MENU
1020 REM ************************
1030 CLS : INK 0: PAPER 7: PRINT
 "          CALCULATOR"
1040 PRINT '"1)INITIALISE"
1050 PRINT '"2)PRINT TABLE"
1060 PRINT '"3)RESET TABLE"
1070 PRINT '"4)DEFINE VARIABLES"
1080 PRINT '"5)DEFINE LINES"
1090 PRINT '"6)DISPLAY COLUMN VA
RIABLES"
1100 PRINT '"7)STOP"
1110 INPUT Z$: CLS
1120 IF Z$="1" THEN GO SUB 1220
1130 IF Z$="2" THEN GO SUB 1300
1140 IF Z$="3" THEN GO SUB 1400
1150 IF Z$="4" THEN GO SUB 1690
1160 IF Z$="5" THEN GO SUB 1810
1170 IF Z$="6" THEN GO SUB 1920
1180 IF Z$="7" THEN GO TO 1200
1190 CLS : GO TO 1030
1200 PRINT AT 10,10;"CALCULATOR"
: INPUT "HAVE YOU ENTERED NEW DA
TA YOU   WISH TO SAVE? (Y/N) ";Q
$: IF Q$="Y" THEN SAVE "CALCULAT
OR": BEEP 1,40: PRINT "REWIND TH
EN ANY KEY TO VERIFY.": PAUSE 0:
 VERIFY "CALCULATOR": PRINT "VER
IFIED"
1210 STOP .
```

Outro módulo de ementa padrão.

MÓDULO 5.1.2

```
1220 REM ************************
1230 REM INITIALISE
1240 REM ************************
1250 DIM A$(20,10): DIM B$(20,10
): DIM C$(20,50): DIM E$(20,50)
1260 DIM B(20,10): DIM D(20,10)
1270 LET LINES=0: LET VARIABLES=
0
1290 RETURN
```

A utilização das várias disposições será explicada no decurso do programa.

MÓDULO 5.1.3

```
1690 REM ************************
1700 REM DEFINE VARIABLES
1710 REM ************************
1720 CLS : FOR I=1 TO VARIABLES:
 PRINT I;") ";B$(I): NEXT I
1730 INPUT "1)EXISTING   2)NEW
 3)QUIT ";Q$: IF Q$="3" THEN RET
URN
1740 IF Q$="1" THEN INPUT "NUMBE
R? ";N: GO TO 1760
1750 IF Q$="2" THEN INPUT "VARIA
BLE NAME? ";N$: LET VARIABLES=VA
RIABLES+1: LET N=VARIABLES: LET
B$(VARIABLES)=N$
1760 CLS : PRINT B$(N): FOR I=1
TO 10: PRINT I;") ";B(N,I): NEXT
 I
```



```
1770 INPUT "NUMBER_""ZZZ"" QUIT_
""DDD"" DELETE  ";Q$: IF Q$="ZZZ
" THEN GO TO 1720
1780 IF Q$="DDD" THEN FOR I=N TO
 VARIABLES-1: LET B$(I)=B$(I+1):
 FOR J=1 TO 10: LET B(I,J)=B(I+1
,J): NEXT J: NEXT I: LET B$(VARI
ABLES)="": FOR I=1 TO 10: LET B(
VARIABLES,I)=0: NEXT I: LET VARI
ABLES=VARIABLES-1: RETURN
1790 INPUT "VALUE? (""X"" EQUAL
TO COLUMN ONE) ";N$: IF N$="X" T
HEN LET B(N,VAL Q$)=B(N,1): GO T
O 1760
1800 LET B(N,VAL Q$)=VAL N$: GO
TO 1760
```

O objectivo deste módulo é permitir ao utilizador designar até vinte variáveis diferentes e especificar uma série de dez valores para cada variável designada, permitindo, deste modo, a fácil realização de cálculos comparativos.

Comentário

Linhas 1720-1730. Os nomes das variáveis definidas até aqui são armazenados na disposição B$ e mostrados ao serem introduzidos neste módulo.

Linha 1750. São colocados na disposição novos nomes de variáveis, numa posição definida pela variável VARIABLES, que é colocada em zero, para começar, e acrescida depois, com cada novo nome.

Linha 1760. Os dez valores associados às variáveis especificadas são armazenados na disposição B, sendo posteriormente mostrados, para que o utilizador possa especificar alterações.

Ensaio do módulo 5.1.3

Neste ponto, as variáveis especificadas não têm qualquer utilidade prática, mas o módulo deverá ser capaz de aceitar até um limite de vinte nomes, juntamente com dez valores associados, e de apagar nomes e valores.

MÓDULO 5.1.4

```
1920 REM ************************
1930 REM DISPLAY COLUMN
1940 REM ************************
1950 CLS : INPUT "WHICH COLUMN?
(0 TO QUIT) ";N: IF N=0 THEN RET
URN
1960 CLS : PRINT "COLUMN ";N
1970 FOR I=1 TO VARIABLES: PRINT
 I;") ";B$(I);" ";B(I,N): NEXT I
1980>INPUT "NUMBER FOR AMENDMENT
_""ZZZ"" QUIT";Q$: IF Q$="ZZZ" T
HEN GO TO 1950
1990 INPUT "NEW VALUE FOR VARIAB
LE? ";R: LET S(VAL Q$,N)=R: GO T
O 1960
```

O módulo anterior permitia a exposição de todos os valores associados a uma única variável. Este módulo mostra o primeiro, segundo ou outro valor associado a cada variável. Assim, se vai ser executada uma série de cálculos sobre uma série de dados plurianuais, este módulo mostrará o valor de cada variável para um determinado ano.

Ensaio do módulo 5.1.4

Se designou algumas variáveis e introduziu alguns valores para elas, deverá poder expor uma coluna de variáveis.

MÓDULO 5.1.5

```
1810 REM ************************
1820 REM DEFINE LINES
1830 REM ************************
1840 CLS : PRINT "              LI
NES"
1850 FOR I=1 TO LINES: PRINT I;"
)";""+A$(I)+"";C$(I): NEXT I
1860 INPUT "1)EXISTING   2)NEW
 3)DELETE   4)QUIT";Q$: IF Q$="4
" THEN RETURN
1870 IF Q$="2" THEN INPUT "LINE
TITLE? ";N$: LET LINES=LINES+1:
LET N=LINES: LET A$(LINES)=N$: I
NPUT "FORMULA? ";F$: LET C$(LINE
S)=F$: GO SUB 1490: LET E$(N)=G$
: GO TO 1840
1880 IF Q$="1" THEN INPUT "NUMBE
R? ";N: INPUT "NEW TITLE? (Y/N)
";T$: IF T$="Y" THEN INPUT "SPEC
IFY NEW TITLE: ";T$: LET A$(N)=T
$
1890 IF Q$="1" THEN INPUT "NEW F
ORMULA? (Y/N) ";T$: IF T$="Y" TH
EN INPUT "SPECIFY NEW FORMULA: "
;F$: LET C$(N)=F$: GO SUB 1490:
LET E$(N)=G$: GO TO 1840
1900 IF Q$="3" THEN INPUT "NUMBE
R? ";N: FOR I=N TO LINES-1: LET
A$(I)=A$(I+1): LET C$(I)=C$(I+1)
: NEXT I: LET A$(LINES)="": LET
C$(LINES)="": LET LINES=LINES-1
1910 RETURN
```

Para entender este módulo precisa de saber alguma coisa quanto às capacidades do *Spectrum* para avaliar uma expressão armazenada numa sequência. Esta parece, à primeira vista, uma ca-

pacidade insignificante. Qual será, eventualmente, a utilidade de se poder instruir o *Spectrum* para imprimir PRINT VAL «1 + 1» — ou qualquer expressão, tanto faz —, quando isso pode ser feito, mais simplesmente, com PRINT 1 + 1? A resposta é que, enquanto uma expressão numérica directa como 1 + 1 apenas pode ser incorporada numa linha programática, «1 + 1» pode ser colocada numa linha programática ou armazenada e manipulada como qualquer outra sequência. Este módulo tira vantagem do facto para armazenar fórmulas introduzidas pelo utilizador.

Comentário

Linha 1850. O número de fórmulas introduzidas até agora é armazenado na variável LINES (linhas). As fórmulas, sob a forma introduzida pelo utilizador, são armazenadas na disposição A$, o que limita o comprimento de cada uma a cinquenta caracteres.

Linha 1870. A fórmula, que pode ter até cinquenta caracteres de comprimento, como foi anteriormente notado, pode ser tudo o que o *Spectrum* reconheça como tendo um valor. O único limite é que os nomes de variáveis não podem conter números. Assim, AA seria reconhecido como o nome de uma variável, enquanto A1 o não seria. A única excepção a isto é que uma fórmula pode utilizar os resultados de outra incluindo uma variável de tipo LINE 1 + 10, o que seria avaliado como a primeira fórmula introduzida mais 10.

Ensaio do módulo 5.1.5

Este módulo não pode ser completamente ensaiado até terem sido introduzidos os dois seguintes. Se for introduzida uma linha temporária, 1490 LET G$ = F$: RETURN, o módulo deverá aceitar fórmulas, juntamente com breves títulos identificativos até dez caracteres e permitir alterações e apagamentos.


```
1490 REM **********************
1500 REM IDENTIFY VARIABLES
1510 REM **********************
1520 LET G$="": LET V$=""
1530 FOR I=1 TO LEN F$: IF F$(I)
=" " THEN GO TO 1610
1540 IF F$(I)>="A" AND F$(I)<="Z
" THEN LET V$=V$+F$(I): GO TO 16
00
1550 IF LEN V$>=4 THEN IF V$(1 TO
4)="LINE" AND (F$(I)> ="0" AND F
$(I)<="9") THEN LET V$ =V$+F$(I)
: GO TO 1600
1560 IF LEN V$>=4 THEN IF V$(1 TO
4)="LINE" AND LEN V$= 4 AND ((F$
(I)<"0" OR F$(I)>"Z") OR (F$(I)<
"A" AND F$(I)>"9")) TH EN PRINT
"LINE UNDEFINED": PAUSE 200:
RETURN
1570 IF LEN V$>=4 THEN IF V$(1
TO 4)="LINE" AND (F$(I)< "A" OR
F$(I)>"Z") THEN GO SUB 16 30:
LET V$="": LET G$=G$+F$(I):GO TO
1600
1580 IF V$<>"" AND (F$(I)<"A" OR
 F$(I)>"Z") THEN GO SUB 1630: LE
T G$=G$+F$(I): LET V$="": GO TO
1600
1590 LET G$=G$+F$(I)
1600 NEXT I
1610 IF LEN V$>0 THEN GO SUB 163
0
1620 RETURN
```

Tendo introduzido algo que o *Spectrum* reconhece como uma fórmula, enfrentamos um problema: o de atribuir um valor às variáveis que a fórmula contém. Ainda que a fórmula contivesse uma

variável com o nome STOTAL, um nome válido para o programa, não poderia extrair-se o valor da fórmula, a menos que o *Spectrum* tivesse um valor para essa variável, o que apenas pode ser feito utilizando um enunciado INPUT ou LET. Por exemplo: LET STOTAL = 10.

Seria difícil definir um tal processo como flexível e fácil de usar, comparado com a forma como foram atribuídos nomes e valores às variáveis, no módulo 3. O facto é que, enquanto pode chamar-se variáveis aos assuntos armazenados em B$, segundo o objectivo do programa, eles nada representam de semelhante para o próprio *Spectrum* e não seriam reconhecidos, se incluídos numa fórmula a ser avaliada. Os nomes de variáveis introduzidos pelo utilizador nunca são usados nos cálculos feitos por este programa, sendo substituídos por outros nomes de variáveis, para os quais o *Spectrum* tem, de facto, um valor acompanhante.

Neste módulo, e no seguinte, a fórmula introduzida pelo utilizador é traduzida em termos que o *Spectrum* possa tratar. Foi já notado que, juntamente com o nome de uma variável, podem ser introduzidos até dez valores, e que estes dez valores ficam armazenados na disposição B. Assim, para o *Spectrum*, cada um destes valores tem um nome de FORM B(X,Y), em que X é o número do nome da variável associado em B$ e Y é o número da coluna de 1 a 10. Estes dois módulos criam uma nova fórmula, que substitui a introduzida pelo utilizador, e cujas variáveis serão elementos tirados da disposição B. Pedir-se ao *Spectrum* que avalie esta segunda fórmula, utilizando a função VAL, visto que todas as suas variáveis estão definidas. Deste modo, uma fórmula introduzida pelo utilizador como CIRCLES*PI*RADIUS12 transformar-se-ia em algo como B(1,3)*PI*B(2,3)12.

Comentário

Linha 1520. G$ será usada para armazenar a fórmula traduzida, à medida que ela for sendo construída, sendo finalmente armazenada na disposição E$, paralela à posição da sua fórmula parente em A$. V$ é utilizada para armazenar, temporariamente, nomes identificados na fórmula introduzida pelo utilizador.

Linha 1530. A fórmula definida pelo utilizador será armazenada numa linha-disposição de cinquenta caracteres, raras vezes preenchendo esse espaço. O módulo considera que a fórmula termina quando encontra o primeiro espaço vazio.

Linha 1540. Se um carácter encontrado na fórmula definida pelo utilizador se incluir dentro dos limites A-Z, será então tomado como fazendo parte do nome de uma variável e acrescentado ao que quer que esteja em V$.

Linha 1550. O primeiro enunciado desta linha deve parecer-lhe um pouco estranho; ele é, de facto, necessário para evitar uma mensagem de erro, caso V$ tenha menos de quatro caracteres de comprimento e seja especificada uma condição tal que IF V$ (1 TO 4) = «LINE». Se V$ tiver menos de quatro caracteres de comprimento, esta linha não será executada; e se V$ tiver quatro ou mais caracteres de comprimento, o enunciado não traz diferença alguma à execução do programa... o que é uma interessante aplicação do modo como o *Spectrum* não cumpre, ao encontrar uma condição falsa. O objectivo da parte principal da linha é estabelecer uma provisão especial para o reconhecimento de variáveis da forma LINE 1, LINE 2, etc., já mencionadas neste livro.

Linha 1560. Esta linha verifica apenas se uma variável começa com as letras LINE e, se assim for, se se segue um número identificativo.

Linha 1570. Esta linha reconhece quando uma variável LINE está completa, através da identificação do primeiro carácter que não possa ser incluído numa tal variável.

Linha 1580. Regressando ao tipo normal de nome de uma variável, que apenas conterá letras, esta linha reconhece quando uma tal variável está completa.

Linha 1590. Os caracteres que não façam parte do nome de uma variável são, simplesmente, acrescentados à fórmula traduzida.

Linha 1610. Normalmente, o complemento do nome de uma variável é reconhecido porque:

A) V$ contém alguns caracteres.
B) O carácter a seguir encontrado não poderia fazer parte de uma variável.

É claro que isto não poderá funcionar se o nome da variável estiver no fim da fórmula, razão pela qual esta linha é utilizada para verificar se V$, esvaziada após cada identificação bem sucedida, continha alguma coisa.

Ensaio do módulo 5.1.6

Apenas poderá realizar-se após o acesso do próximo módulo reduzido.

MÓDULO 5.1.7

```
1630 REM ************************
1640 REM EVALUATE
1650 REM ************************
1660 IF LEN V$>=4 THEN IF V$(1
TO 4)="LINE" THEN LET G$ =G$+"D
("+V$(5 TO )+",J)": RETURN
1670 FOR J=1 TO VARIABLES: IF V$
<>B$(J,1 TO LEN V$) THEN NEXT J:
 PRINT "VARIABLE ";V$;" NOT FOUN
D.": PAUSE 200: LET IND=0: RETUR
N
1680 LET G$=G$+"B("+STR$ J+",J)"
: RETURN
```

Depois de o módulo anterior ter identificado nomes da fórmula definida pelo utilizador, este módulo traduz esses nomes em termos que o *Spectrum* possa entender nos seus cálculos.

Comentário

Linha 1660. Os resultados de avaliações de fórmulas serão colocados na disposição D, posteriormente. Esta linha traduz o nome de uma variável como LINE3 para o nome de um elemento em D, como o D(3,J). «J» é utilizado porque, posteriormente, a fórmula armazenada será avaliada por dois desvios: um desvio I, referido ao número de fórmulas, e um desvio J, referido a cada um dos dez valores possíveis para cada variável da linha 1670. Para uma variável vulgar, esta linha verifica, através dos nomes de variáveis existentes, se não foi definida uma tal variável, informando o utilizador do resultado.

Linha 1680. Se o nome da variável foi já declarado, o elemento relevante da disposição B é acrescentado a G$. Se a variável STOTAL era terceira na lista de variáveis, será transferida como B(3,J).

Ensaio do módulo 5.1.7

Deverá agora poder introduzir os nomes das variáveis e especificar uma série de valores para cada. Introduza uma fórmula composta por números, símbolos matemáticos e variáveis que tenha definido. Pare o programa e verifique se isto foi traduzido razoavelmente. Deverá encontrar este material armazenado na sequência temporária G$ e, como elemento, na disposição E$. Se você conhece a resposta correcta para a sua fórmula específica, pode ensaiar a fidelidade dos módulos introduzindo PRINT VAL G$ em modo directo. Deverá também poder utilizar os resultados de uma fórmula como variável noutra, utilizando variáveis LINE.

MÓDULO 5.1.8

```
1400 REM ************************
1410 REM RESET TABLE
1420 REM ************************
1430 INPUT "HOW MANY COLUMNS? ";
```

```
COLUMNS
1440 FOR K=1 TO LINES: LET F$=C$
(K): GO SUB 1490: LET E$(K)=G$:
NEXT K
1450 FOR I=1 TO LINES: FOR J=1 T
O COLUMNS
1460 LET D(I,J)=VAL E$(I)
1470 NEXT J: NEXT I
1480 RETURN
```

Após ter avaliado os resultados das fórmulas definidas pelo utilizador, utilizando a gama de valores especificada para as variáveis denominadas, poderá desejar alterar uma certa variável. Em vez de reavaliar tudo, de cada vez que se altera uma variável, este módulo recalcula todo o quadro de «linhas e colunas» sempre que o utilizador o especifique. Fórmulas não traduzidas são solicitadas à disposição C$ e retraduzidas, com base nos últimos dados.

Comentário

Linha 1460. Pode parecer, à primeira vista, que esta linha coloca uma série completa de elementos da disposição D no mesmo valor, o que não é verdade. A avaliação de E$(I) alterar-se-á com qualquer mudança em J, uma vez que já transferimos as variáveis definidas pelo utilizador para a forma B(X,J) ou D(X,J). Note-se que não é feita qualquer tentativa de tratar os problemas levantados com a interacção dos valores das linhas (LINE). Se o valor de LINE1 depender do valor de LINE2 e LINE2 for alterada por este desvio, o valor de LINE1 não terá em conta esse facto, a não ser que o módulo seja solicitado duas vezes. Interacções mais complexas entre estas linhas devem ser ponderadas, antes de serem introduzidas.


```
1300 REM ************************
1310 REM PRINT TABLE
1320 REM ************************
1330 INPUT "COLUMN FOR DISPLAY?
(0=QUIT)";COLUMN: PAPER 7: IF CO
LUMN=0 THEN PAPER 7: RETURN
1340 CLS : PRINT "LINES       FIG
URES"
1350 FOR I=1 TO LINES: PAPER (5+
2*(I/2=INT (I/2))): PRINT A$(I);
" ";
1360 LET M$=STR$ D(I,COLUMN): IF
 LEN M$>10 THEN PRINT M$( TO 10)
: GO TO 1380
1370 PRINT M$
1380 NEXT I
1390 GO TO 1330
```

Este módulo imprime no *écran* o breve título de cada linha, ou fórmula, e o valor derivado da linha, quando baseado numa das dez colunas de valores de variáveis.

Comentário

Linhas 1360-1370. Estas duas linhas limitam qualquer imagem a dez dígitos. O único objectivo disto é permitir a exposição de duas colunas, com alterações mínimas ao programa. Se for provável que venha a ter dez dígitos ou mais, com frequência, é natural que venha a querer alterar isto.

Ensaio do módulo 5.1.9

Deverá agora poder ensaiar integralmente o programa, introduzindo até vinte variáveis, cada uma com dez valores prováveis, até um limite de vinte linhas de fórmulas, utilizando estas variáveis e breves títulos para cada uma destas linhas, expondo depois os resultados, coluna a coluna. Se estes testes forem satisfatórios, o programa estará pronto a ser utilizado.

Sumário

Pelo modo como se apresentam os programas de aplicação múltipla, este parece insípido, à primeira vista. No entanto pode tornar-se altamente útil numa grande variedade de aspectos. Nos negócios, pode ser utilizado para examinar os efeitos de alterações no custo de materiais ou preços, ou ainda em ambos. Em casa, pode servir de adjunto aos programas financeiros do cap. 2, levando a cabo todos os cálculos referentes a impostos, se você conseguir organizar a fórmula aplicável. Os amadores podem desejar utilizá-lo em cálculos repetitivos, noutras áreas. Tal como muitos dos programas deste livro, este é uma ferramenta flexível e você apenas terá a ganhar em jogar com ele, aplicá-lo e alterá-lo de acordo com as suas necessidades, porquanto isso lhe permitirá uma correcta visão intrinseca das capacidades do programa.

Avançando mais

1) Caso você não precise de mais de três ou quatro dígitos nas suas aplicações, adapte o programa de forma a que ele possa exibir várias colunas ao mesmo tempo.

2) Se você possuir um *Spectrum* de 48K, alargue o programa de modo a cobrir mais variáveis e fórmulas, criando assim a oportunidade de elaborar uma biblioteca de fórmulas úteis.

5.2 CALORIAS

Tal como se apresenta, este programa pode, ou não, ser-lhe útil. Se o que você precisa é de contar as calorias, então ele pode poupar-lhe muito tempo e tornar mais precisos os seus esforços nesse sentido. Quer você esteja interessado, ou não, em calorias, este programa inclui um estudo cuidadoso, como exemplo do modo segundo o qual o estilo modular de programação que adoptámos permite a adaptação rápida a outras utilizações de material previamente escrito. Este programa é uma cópia tenuemente disfarçada de vastas secções do «MultiQ». A brevidade dos comentários sobre o programa dar-lhe-á uma certa ideia da rapidez com que ele foi escrito. Apesar desse facto, trata-se de um programa importante, no sentido em que demonstra como os dicionários de assuntos e de quantidades relativas podem ser elaborados e utilizados.

MÓDULO 5.2.1

```
1180 REM ************************
1190 REM MENU
1200 REM ************************
1210 BORDER 4: INK 0: PAPER 7: C
LS : PRINT "                MENU"
1220 PRINT '"1)DISPLAY TODAY'S L
IST"
1230 PRINT '"2)INPUT TO TODAY'S
LIST"
1240 PRINT '"3)START FRESH LIST"
1250 PRINT '"4)DELETE FROM DAY'S
 LIST"
1260 PRINT '"5)EXTEND DICTIONARY
"
1270 PRINT '"6)DISPLAY DICTIONAR
Y/DELETE"
1280 PRINT '"7)INITIALISE"
1290 PRINT '"8)STOP"
1300 INPUT "Which do you require
?";Z$: CLS
```

```
1310 IF Z$="1" THEN GO SUB 1430
1320 IF Z$="2" THEN GO SUB 1580
1330 IF Z$="3" THEN GO SUB 1700
1340 IF Z$="4" THEN GO SUB 2520
1350 IF Z$="5" THEN GO SUB 1740
1360 IF Z$="6" THEN GO SUB 2170
1370 IF Z$="7" THEN GO TO 1030
1380 IF Z$="8" THEN GO TO 1400
1390 CLS : GO TO 1180
1400 PRINT FLASH 1;AT 10,12;"CAL
ORIES"
1410 INPUT "HAVE YOU INPUT NEW I
NFORMATION  YOU WISH TO SAVE? (Y
/N)";Q$: IF Q$="Y" THEN SAVE "CA
LORIES": BEEP 1,40: PRINT '"REWI
ND, THEN ANY KEY TO VERIFY": PAU
SE 0: VERIFY "CALORIES": PRINT '
"VERIFIED"
1420 STOP
```

Um módulo de ementa padrão.

MÓDULO 5.2.2

```
   1 GO TO 1180
1000 REM ************************
1010 REM VARIABLES
1020 REM ************************
1030 DIM A$(500,15)
1040 DIM B$(500,15)
1050 DIM C(500)
1060 DIM T$(50,2,20)
1070 DIM Q$(32)
1080 DIM T(30)
1090 LET LIST=0
```



```
1100 LET ENDITEM=3
1110 LET ITEMS=2
1120 LET I$=CHR$ 0+CHR$ 1+CHR$ 0
+CHR$ 2
1130 LET A$(2,1)=CHR$ 255
1140 LET P$="     "
1150 DIM F$(15)
1160 DIM G$(17)
1170 LET L$="******************
**********"
```

Por que razão coloquei as variáveis antes da ementa? Bem, na altura, pareceu-me uma boa ideia. A utilização das várias disposições será discutida no decurso do programa.

MÓDULO 5.2.3

```
1740 REM ************************
1750 REM INPUT ITEMS
1760 REM ************************
1770 CLS : PRINT "   NEW ITEMS F
OR DICTIONARY"
1780 PRINT '"FOOD:": INPUT "NAME
 OR ""ZZZ"" TO QUIT";F$: IF F$(
TO 3)="ZZZ" THEN RETURN
1790 PRINT AT 2,5;F$
1800 PRINT '"UNITS:": INPUT "NAM
E ";G$
1810 PRINT AT 4,6;G$
1820 PRINT '"CALORIES PER UNIT:"
: INPUT "NAME ";H$
1830 LET TEMP=VAL H$
1840 PRINT AT 6,18;H$
1850 INPUT "Are these correct? (
Y/N)";Q$: IF Q$="N" THEN GO TO 1
770
```

```
1860 CLS : GO SUB 1880: GO SUB 2
030
1870 LET ITEMS=ITEMS+1: GO TO 17
40
```

Este módulo directo aceita três inserções, representando o nome de um alimento, as unidades em que ele é habitualmente medido e o número de calorias por unidade.

MÓDULO 5.2.4

```
1880 REM ************************
1890 REM BINARY SEARCH
1900 REM ************************
1910 DEF FN A( )=256*CODE I$(2*S-
1)+CODE I$(2*S)
1920 LET POWER=INT (LN (ITEMS)/L
N 2)
1930 LET S=2^POWER
1940 FOR I=POWER-1 TO 0 STEP -1
1950 LET P=FN A( )
1960 LET S=S-(2^I)*(A$(P)>F$)+(2
^I)*(A$(P)<F$)
1970 IF S<2 THEN LET S=2
1980 IF S>ITEMS-1 THEN LET S=ITE
MS-1
1990 NEXT I
2000 LET P=FN A( )
2010 IF A$(P)>F$ THEN LET S=S-1
2020 RETURN
```

Trata-se de uma busca binária directa, para determinar o local do alimento no conjunto do dicionário de alimentos, armazenado em **A$**. Podem ser armazenados até quinhentos alimentos, cada um deles com um nome até quinze caracteres de comprimento.

MÓDULO 5.2.5

```
2030 REM ************************
2040 REM INSERTION
2050 REM ************************
2060 IF LEN P$=4 THEN GO TO 2100
2070 LET PLACE=256*CODE P$(3)+CO
DE P$(4)
2080 LET P$=P$( TO 2)+P$(5 TO )
2090 GO TO 2120
2100 LET PLACE=ENDITEM
2110 LET ENDITEM=ENDITEM+1
2120 LET A$(PLACE)=F$
2130 LET B$(PLACE)=G$
2140 LET C(PLACE)=TEMP
2150 LET I$=I$( TO 2*S)+CHR$ INT
 (PLACE/256)+CHR$ INT (PLACE-256
*INT (PLACE/256))+I$(2*S+1 TO )
2160 RETURN
```

Este módulo insere o novo assunto no local correcto, segundo a ordem praticada no método salientado no «MultiQ». Em primeiro lugar são preenchidos os espaços vazios da lista, seguindo-se as posições no final da lista. A verdadeira ordem é indicada por pares de caracteres na disposição para indicadores, em **I$**.

Ensaio do módulo 5.2.5

Neste ponto, embora o dicionário não possa facilmente ser exibido, você deverá ser capaz de introduzir assuntos no dicionário, os quais serão armazenados em **A$**, pela ordem em que foram introduzidos. Unidades associadas e valores calóricos deverão ser armazenados nas posições paralelas, em B$ e C, respectivamente. A ordem correcta dos assuntos deverá ser discernível a partir do valor de código de pares de caracteres em **I$**.

MÓDULO 5.2.6

```
2170 REM ************************
2180 REM USER SEARCH
2190 REM ************************
2200 PRINT "                DISPLA
Y"
2210 PRINT '"ON APPEARANCE OF IT
EM, INPUT:"
2220 PRINT '">POSITIVE OR NEGATI
VE NUMBER TO MOVE POINTER"
2230 PRINT '">""DDD"" TO DELETE
ITEM"
2240 PRINT '">""ZZZ"" TO QUIT FU
NCTION"
2250 PRINT L$
2260 IF ITEMS=2 THEN RETURN
2270 LET S=2
2280 LET P=FN A( )
2290 PRINT AT 12,0;"ENTRY No. ";
S-1
2300 PRINT '"FOOD:";A$(P)
2310 PRINT '"UNITS:";B$(P)
2320 PRINT '"CALORIES:";C(P);"
"
2330 INPUT "Which do you require
?";S$
2340 IF S$="DDD" THEN GO SUB 244
0: RETURN
2350 IF S$="ZZZ" THEN RETURN
2360 IF S$<>"" THEN GO TO 2400
2370 LET S=S+1
2380 IF S=ITEMS THEN RETURN
2390 GO TO 2280
2400 LET S=S+VAL S$
2410 IF S>=ITEMS THEN RETURN
2420 IF S<2 THEN LET S=2
2430 GO TO 2280
```


Trata-se de um simples módulo de busca, baseado no utilizado no «MultiQ». Não existe uma efectiva busca; o utilizador apenas pode rever para trás e para a frente. O módulo de apagamento é igualmente solicitado por este módulo.

Ensaio do módulo 5.2.6

Deverá agora poder exibir todos os alimentos introduzidos, à ordem.

MÓDULO 5.2.7

```
2440 REM ************************
2450 REM DELETIONS
2460 REM ************************
2470 LET P=FN A( ): LET A$(P)="":
 LET B$(P)="": LET C(P)=0
2480 LET I$=I$( TO 2*S-2)+I$(2*S
+1 TO )
2490 LET ITEMS=ITEMS-1
2500 LET P$=P$( TO 2)+CHR$ INT (
P/256)+CHR$ INT (P-256*INT (P/25
6))+P$(3 TO )
2510 RETURN
```

Trata-se do método de apagamento utilizado no «MultiQ». O endereço do assunto apagado é removido de I$ e o espaço vazio deixado é registado em P$. Note-se que o acesso não é, de facto, removido em ponto algum da disposição. O programa apenas deixa de o ter em conta.

Ensaio do módulo 5.2.7

Deverá agora poder apagar assuntos do dicionário.

MÓDULO 5.2.8

```
1580 REM ************************
1590 REM ADD TO TODAY'S LIST
1600 REM ************************
1610 PRINT "      ADD TO TODAY'S
LIST"
1620 PRINT '"FOOD:": INPUT F$: P
RINT AT 2,5;F$
1630 GO SUB 1880: IF A$(P)<>F$ T
HEN PRINT AT 10,0;"*SPECIFIED  F
OOD UNKNOWN, PLEASE CHECK.": PAU
SE 200: RETURN
1640 PRINT '"UNITS:";B$(P)
1650 PRINT '"QUANTITY:": INPUT Q
: PRINT AT 6,9;Q
1660 INPUT "Are these correct? (
Y/N)";Q$: CLS : IF Q$="N" THEN G
O TO 1580
1670 LET LIST=LIST+1: LET T$(LIS
T,1)=A$(P): LET T$(LIST,2)=STR$
Q+" "+B$(P): LET T(LIST)=Q*C(P)
1680 INPUT ."Any more items? (Y/N
)";Q$: CLS : IF Q$="Y" THEN GO T
O 1580
1690 RETURN
```

Com este módulo afastamo-nos do enquadramento do «MultiQ» e atingimos as partes exclusivas da nossa nova aplicação. O objectivo deste módulo é permitir a rápida criação de uma lista diária de alimentos, baseada em alimentos já contidos no dicionário. Não é necessário que a lista seja feita de uma só vez, antes à medida que forem consumidos alimentos ou planeadas ementas.

Linha 1630. Note-se o interessante uso do módulo de busca binária nesta linha. Ao ser introduzido um alimento para a lista diária, o respectivo nome é enviado para o módulo de busca, que identifica a posição que o alimento teria, caso estivesse contido no dicionário. Ao regressar da busca, o alimento nessa posição é comparado com o alimento introduzido na lista diária. Se não forem o mesmo alimento, aquele que foi introduzido na lista diária não se encontra no dicionário, sendo o utilizador informado do facto. A utilização de RANDOMIZE no enunciado 2 é apenas um meio de obviar ao erro, caso o alimento seja encontrado no dicionário.

Linha 1670. A lista diária corrente é armazenada em duas metades da disposição T$ e os valores calóricos associados em T.

Ensaio do módulo 5.2.8

O programa deverá agora aceitar a introdução de alimentos na lista diária, exibir as unidades de medida habituais e solicitar uma introdução de quantidades. Não serão aceites alimentos que não estejam incluídos no dicionário.

MÓDULO 5.2.9

```
1430 REM ************************
1440 REM TODAY'S LIST
1450 REM ************************
1460 LET TOTAL=0
1470 FOR I=1 TO LIST
1480 PRINT "FOOD     :";T$(I,1)
1490 PRINT "AMOUNT   :";T$(I,2)
1500 PRINT "CALORIES:";T(I)
1510 PRINT L$
1520 LET TOTAL=TOTAL+T(I)
1530 NEXT I
1540 PRINT '"TOTAL CALORIES:";TO
TAL
1550 PRINT '"Press any key to co
ntinue."
1560 PAUSE 0
1570 RETURN
```

Este módulo apenas exibe a lista de alimentos para o dia, dando os respectivos nomes, a quantidade e o total de calorias para essa quantidade. No final da lista é exibido o total de calorias da lista.

MÓDULO 5.2.10

```
1700 REM ************************
1710 REM NEW LIST
1720 REM ************************
1730 DIM T$(50,2,20): DIM T(50):
 LET LIST=0: GO SUB 1580: RETURN
```

Este módulo permite a indicação das disposições relativas à lista diária, sem que isso afecte as disposições principais.

MÓDULO 5.2.11

```
2520 REM ************************
2530 REM DELETE FROM LIST
2540 REM ************************
2550 FOR I=1 TO LIST
2560 PRINT T$(I,1)
2570 INPUT "DDD=DELETE/ENTER=NEX
T/ZZZ=QUIT";Q$: IF Q$="ZZZ" THEN
 RETURN
2580 IF Q$="DDD" THEN GO TO 2600
2590 NEXT I: RETURN
2600 LET T$(I,1)="": LET T$(I,2)
="": LET T(I)=0
2610 FOR J=I TO LIST-1
2620 LET T$(J,1)=T$(J+1,1)
2630 LET T$(J,2)=T$(J+1,2)
2640 LET T(J)=T(J+1)
2650 NEXT J
2660 LET LIST=LIST-1
2670 RETURN
```


Este módulo apaga assuntos da lista diária.

Ensaio do módulo 5.2.11

Deverá agora poder organizar uma lista diária, exibi-la e apagar partes dela. Deverá também poder indicar as disposições para começar uma lista nova. Se estes ensaios forem satisfatórios, o programa estará pronto a ser utilizado.

Sumário

Espero que, no decurso deste programa, se tenha apercebido das vantagens do método modular na feitura de programas. Ele permitiu à maior parte deste programa ser retirada de uma aplicação muito diferente. À medida que o tempo passe e que você vá desenvolvendo os seus métodos preferidos para tratar com diferentes tipos de materiais e processos, desenvolver-se-á igualmente uma sensibilidade para formas de conseguir novas aplicações para o seu *Spectrum*, utilizando métodos que já compreende e linhas programáticas jás ensaiadas e desparasitadas.

Daqui, deve você tirar a importante lição de que é mais importante reunir métodos do que programas, embora os programas também possam oferecer, muitas vezes, novos métodos. Uma boa biblioteca de programas ser-lhe-á vantajosa até aparecerem novas aplicações. Uma boa colecção de métodos, contidos em módulos funcionais, constitui uma útil base de apoio, seja qual for o objectivo a atingir. Procure nos livros e revistas existentes no mercado novas formas de realizar tarefas e, logo que encontre uma boa, incorpore-a num programa funcional, ainda que não seja realmente vital. Ficará a aguardar os dias cinzentos.

No meio de todas estas generalizações, não perca de vista o facto de este programa ter uma vasta gama de potenciais aplicações. De forma alguma os alimentos constituem a única área onde se revela necessária a existência de um dicionário de assuntos, associados às suas unidades típicas e um qualquer valor. Os livreiros, por exemplo, quase sempre armazenam menos de quinhentas linhas, cada uma delas vendida numa unidade de qualquer espécie, por um determinado preço. Por que não adaptar o programa para

alguns destes usos? Lembre-se que são os métodos que contam, mais do que os programas. Este programa é um exemplo funcional de um método, se entende o que eu estou a dizer...

Avançando mais

1) Uma diferença fundamental entre este programa e o «MultiQ» é que este último dimensionou as suas próprias disposições para fazer frente a uma variedade de aplicações, embora utilizando a memória até ao máximo. Poderá alterar o programa, de modo a que ele se torne um utensílio para nomes e quantidades de aplicação geral, com os nomes dos assuntos especificados pelo utilizador?!

5.3 GRÁFICO

Este curto programa é um desenhador de gráficos de aplicação geral, que permite ao utilizador definir as unidades em ambos os eixos, tanto em termos de nome como de extensão, e introduzir dados, por ordem ou não, para criar um gráfico linear.

MÓDULO 5.3.1

```
1000 REM ************************
1010 REM MENU
1020 REM ************************
1030 CLS : PRINT AT 0,12; FLASH
1; INK 2;"GRAPH"
1040 PRINT '"1)SET UP FRAMEWORK"
1050 PRINT '"2)INPUT VALUES"
1060 PRINT '"3)DRAW GRAPH"
1070 PRINT '"4)INITIALISE"
1080 PRINT '"5)STOP"
1090 INPUT Z$: CLS
1100 IF Z$="1" THEN GO SUB 1180
1110 IF Z$="2" THEN GO SUB 1350
1120 IF Z$="3" THEN GO SUB 1530
1130 IF Z$="4" THEN CLEAR : LET
Z$="1"
1140 IF Z$="5" THEN GO TO 1160
1150 GO TO 1000
1160 PRINT AT 10,12; FLASH 1; IN
K 2;"GRAPH": INPUT "DO YOU WISH
TO RECORD?";Q$: IF Q$="Y" THEN S
AVE "GRAPH": INPUT "REWIND, THEN
 ""ENTER"" TO VERIFY.";Q$: VERIF
Y "GRAPH": PRINT '"      PROGRAM
VERIFIED."
1170 STOP
```

Outro módulo de ementa padrão. Saliente-se a linha 1130: Z$ precisa de ser colocado em «1», após ter limpo (CLEAR) a memória ou criado uma mensagem de erro na linha 1140, uma vez que não haverá aí qualquer Z$.

MÓDULO 5.3.2

```
1180 REM ************************
1190 REM SET UP AXES
1200 REM ************************
1210 PRINT "HOW MANY INTERVALS O
N THE       HORIZONTAL AXIS? ";:
 INPUT H: PRINT H: LET LH=INT (2
36/H)
1220 PRINT "HOW MANY INTERVALS O
N THE       VERTICAL ACCESS? ";:
 INPUT V: PRINT V: LET LV=INT (1
56/V)
```

```
1230 CLS : PRINT AT 8,3; FLASH 1
; INK 2;"AXES WILL LOOK LIKE THI
S"
1240 LET VMARK=300: LET HMARK=30
0: GO SUB 1450
1250 INPUT "YOU CAN SET A MARKER
 EVERY SO    MANY SPACES.
             INPUT GAP FOR HORIZ.
:";HMARK
1260 INPUT "GAP FOR VERTICAL:";V
MARK
1270 CLS : GO SUB 1450
1280 INPUT "IS THAT SATISFACTORY
? (Y/N)";Q$: IF Q$="N" THEN CLS
: GO TO 1180
1290 INPUT "NAME OF HORIZ. UNITS
? ";H$
1300 INPUT "NAME OF VERT. UNITS?
 ";V$
1310 PRINT '"MIN. VALUE ON V. AX
IS? ": INPUT BASE
1320 PRINT "MAX. VALUE REPRESENT
ED BY ";V'"UNITS? ": INPUT LIMIT
1330 LET VVAL=(LIMIT-BASE)/V
1340 DIM G(H): FOR I=1 TO H: LET
 G(I)=-999: NEXT I: RETURN
```

O objectivo deste módulo é permitir ao utilizador especificar quantos intervalos haverá nos eixos horizontal e vertical, e atribuir um nome às unidades envolvidas.

Comentário

Linha 1210. O eixo horizontal terá 236 *pixels* de comprimento. A variável LH (comprimento horizontal) é colocada em 236, divididos pelo número de intervalos desejado. Regra geral, isto resultará num número de intervalos ligeiramente superior ao especificado pelo utilizador.

Linha 1220. É criada uma variável similar para o eixo vertical.

Linhas 1240-1280. As unidades serão marcadas nos eixos por pequenas linhas desenhadas (DRAW). Como ajuda à maior clareza destas linhas, algumas delas podem ser ligeiramente mais compridas. Por exemplo, cada quinta unidade pode ser avivada. Os intervalos a que isto ocorrerá ficam armazenados em HMARK e VMARK, sendo ambas fixadas em 300, logo que os eixos sejam desenhados pela primeira vez. Desta forma, nenhuma marca se salientará das outras, em qualquer dos eixos. Assim que os eixos básicos tenham sido exibidos, o utilizador é solicitado a especificar os intervalos dos pontos a serem intensificados e os eixos reexibidos, para confirmação.

Linhas 1290-1320. São solicitados os nomes das unidades em ambos os eixos. Parte-se do princípio de que as unidades do eixo horizontal começam do zero mas, para o eixo vertical, pede-se ao utilizador que especifique o valor em que começa o eixo e o valor em que acaba. A partir daqui é calculado o valor de uma unidade individual no eixo vertical, o qual é guardado na variável VVAL. Parte-se do princípio de que as unidades no eixo horizontal têm o valor 1.

Linha 1340. A disposição G é dimensionada, pronta a receber os possíveis dados de assuntos H (horizontais).

Ensaio do módulo 5.3.2

Os eixos não podem ser desenhados até que o próximo módulo seja introduzido, a não ser inserindo uma linha temporária, 1450 RETURN. O módulo deverá poder solicitar o número de intervalos, a marca a ser intensificada e o nome das unidades.

```
1450 REM **********************
1460 REM DRAW AXES
1470 REM **********************
1480 FOR I=1 TO 19: PRINT AT I,1
;"▌": NEXT I: PRINT AT 20,1;"▙";
1490 FOR I=1 TO 29: PRINT "▄";:
NEXT I
1500 LET C=0: FOR I=11+LH TO 247
 STEP LH: LET C=C+1: PLOT INVERS
E 1; OVER 1;I,13: DRAW OVER 1;0,
-5-3*(C/HMARK=INT (C/HMARK)): NE
XT I
1510 LET C=0: FOR I=11+LV TO 167
 STEP LV: LET C=C+1: PLOT INVERS
E 1; OVER 1;13,I: DRAW OVER 1;-5
-3*(C/VMARK=INT (C/VMARK)),0: NE
XT I
1520 RETURN
```

Este módulo desenha os eixos do gráfico.

Comentário

Linha 1500. O eixo horizontal é preenchido com linhas de intersecção de cinco *pixels* de comprimento e separadas LH *pixels*. Por cada intervalo HMARK, a linha aumenta três *pixels*.

Linha 1510. Repete-se um processo similar para o eixo vertical.

Ensaio do módulo 5.3.3

Deverá agora poder introduzir pormenores de unidades e nomes de unidades e expor os eixos, para confirmação.


```
1350 REM **********************
1360 REM INPUT VALUES
1370 REM **********************
1380 PRINT "NUMBER OF ";H$;"? ";
: INPUT H1: PRINT H1
1390 PRINT '"NUMBER OF ";V$;"? "
;: INPUT V1: PRINT V1
1400 PRINT '"ARE THESE CORRECT?"
: INPUT Q$: IF Q$="N" THEN CLS :
 GO TO 1380
1410 IF G(H1)<>-999 THEN PRINT '
"THAT POSITION IS ALREADY
 OCCUPIED BY THE VALUE ";G(H1);"
."'"DO YOU WISH TO REPLACE IT? (
Y/N)": INPUT Q$: IF Q$="N" THEN
CLS : GO TO 1350
1420 LET G(H1)=V1: INPUT "ANOTHE
R VALUE?";Q$: IF Q$="Y" THEN CLS
 : GO TO 1350
1430 RETURN
```

Este módulo aceita os dados que serão utilizados para desenhar o próprio gráfico.

Comentário

Linha 1410. Quando a disposição G foi organizada, foi preenchida com o valor —999, para indicar elementos vagos, uma vez que é pouco provável a utilização frequente de 0. Se um assunto for introduzido para uma posição que ainda não esteja preenchida com —999, o utilizador será informado de que já foram introduzidos dados para essa posição.

Linha 1420. Os dados são colocados na disposição G, não sendo feita qualquer tentativa para verificar se eles se incluem dentro dos limites especificados para o gráfico, embora o utilizador fosse convidado a confirmar os dados, na linha 1400. Note-se que, uma vez que se presume que as unidades do eixo horizontal vão evoluir em unidades de 1, a partir de uma base de 1, a posição de um assunto no eixo horizontal é a mesma que em G.

Ensaio do módulo 5.3.4

O gráfico não pode ser desenhado ainda, embora o módulo deva aceitar dados, informando-o se está a escrever dados já presentes.

MÓDULO 5.3.5

```
1530 REM **********************
1540 REM DRAW GRAPH
1550 REM **********************
1560 PRINT AT 0,0;V$;" BASE:";BA
SE;"/LIMIT:";LIMIT: FOR I=1 TO L
EN H$: PRINT AT 21-LEN H$+I,31;H
$(I): NEXT I
1570 GO SUB 1450
1580 FOR I=1 TO H: IF G(I)=-999
THEN NEXT I: PRINT AT 10,3;"NO D
ATA.": STOP
1590 PLOT INVERSE 1; OVER 1;11+I
*LH,11+(G(I)-BASE)/VVAL*LV
1600 FOR J=I TO H: IF GGG(J)=-999
THEN GO TO 1640
1610 FOR K=J+1 TO H
1620 IF G(K)=-999 THEN NEXT K: I
F K>H THEN GO TO 1650
1630 DRAW LH*(K-J),(G(K)-G(J))/V
VAL*LV
1640 NEXT J
1650 INPUT """ENTER"" TO CONTINU
E";Q$: RETURN
```


Este módulo desenha o verdadeiro gráfico, baseado nos dados fornecidos pelo utilizador.

Comentário

Linha 1560. Os nomes das unidades para os eixos horizontal e vertical são impressos em locais apropriados.

Linha 1580. À primeira vista, esta linha aceita alguns esclarecimentos. O que ela faz, de facto, é localizar um ponto invisível onde o gráfico deverá começar. 11 + I*LH é a posição horizontal correspondente ao primeiro assunto dos dados, na disposição G. (G(I) — BASE)/VVAL é o número de unidades que isto deve representar no eixo vertical. (G(I) — BASE)/VVAL*LV é a altura, em *pixels*, do ponto determinado.

Linhas 1600-1640. São necessários dois desvios: um para registar a posição do último ponto localizado; o outro para encontrar a posição seguinte a ser utilizada. Não é necessário que o utilizador preencha todos os espaços de dados, pelo que o módulo omitirá os elementos vazios em G. Uma vez encontrado um item de dados, a linha 1630 desenha uma linha até um ponto correspondente, horizontalmente, à posição dos dados em G, tendo em conta todos os espaços vazios que encontre pelo caminho, e, verticalmente, ao número de *pixels* acima ou abaixo do valor previamente localizado. (G(K) — G(J))/VVAL*LV é directamente paralelo à expressão explicada na linha 1590.

Ensaio do módulo 5.3.5

Se já introduziu dados razoáveis, deverá agora estar em posição de desenhar um gráfico. Se tal for conseguido satisfatoriamente, o programa estará pronto a ser utilizado.

Sumário

É pouco provável que muitos utilizadores do *Spectrum* queiram usar um programa como este todos os dias. Mas é um exemplo do tipo de utensílio que pode ser validamente acrescentado a uma

biblioteca de programas, sabendo que, um dia, quererá desenhar um gráfico, embora não com tanto empenho que justifique a elaboração de um programa especial. Poderá também ter notado, ao introduzir o programa, que, ainda que seja pouca coisa a adição de pequenos retoques, como o da intensificação de certos pontos dos eixos, a exibição de nomes de unidades e o ponto de partida do eixo vertical, estes pormenores podem marcar uma diferença entre um gráfico que apenas está correcto e um outro que esteja compreensível e correcto. Afinal, qual a vantagem de se armazenar algo, se quando voltamos a olhar para aquilo que armazenámos não nos conseguimos lembrar do que isso representa ou do que são as unidades?

Avançando mais

1) Um projecto bastante simples é o de adaptar o programa de modo a que ele também desenhe gráficos de barras. As barras poderiam ter espessuras variáveis, de aplicação para aplicação, de forma que a única alteração ao programa seria a do módulo que desenha o gráfico. Tendo estabelecido o valor numa determinada posição, o módulo teria de desenhar linhas LH até abaixo, à base horizontal, ou talvez LH—1, de forma a fazer-se uma separação entre as barras.

2) Seria útil que o programa tivesse um dispositivo para mostrar o gráfico sem parar o programa, ao introduzir COPY em modo directo.

5.4 RENUMERAÇÃO

Depois de ter sugerido que a maior parte dos utilizadores do *Spectrum* raramente usariam o programa anterior todos os dias, passarei ao extremo oposto, para este programa, e direi que o utilizareis, provavelmente, mais do que qualquer outro da vossa biblioteca. Para aqueles que não estão familiarizados com a ideia, um programa de renumeração é aquele que altera os números de linha irregulares que se vão desenvolvendo à medida que a maioria dos

programas vão sendo escritos e os insere na estrutura regular vista na maior parte dos programas deste livro.

Esta tarefa não é tão difícil como pode parecer. Os números de linha são armazenados no princípio de cada linha, em dois *bytes*... da mesma forma que já usámos para armazenar valores, em dois caracteres incluídos em sequências. Tudo o que é preciso fazer é encontrar a posição do número de linha que inicia o programa, meter (POKE) nas duas posições o número de linha com que se quer começar e passar pelos números de linha metendo números que vão aumentando, a intervalos regulares. Este trabalho pode ser feito em três ou quatro linhas.

O problema é que a renumeração não fica por aí, devido à pequena questão dos GOTO e GOSUB. Espalhadas ao longo de todo o programa encontrará referências aos números de linha originais. Se esses números de linha foram alterados, gerar-se-á o caos. Assim, é preciso renumerar GOTO e GOSUB que condigam com os números de linha alterados.

Isto não é fácil. A forma como os destinos GOTO e GOSUB são gravados é consideravelmente diferente e mais complexa do que a forma como os números de linha são armazenados. Já antes salientámos que, normalmente, os valores numéricos requerem cinco *bytes* da memória do *Spectrum*, para contar com a vasta gama de números que o *Spectrum* pode expressar com rigor. Em alguns aspectos, infelizmente, os destinos GOTO e GOSUB empregam esta forma mais complexa de armazenamento, apesar de os 9999 números de linha possíveis apenas necessitarem, na realidade, de dois caracteres para gravar.

Em certos aspectos, isto aumenta, de facto, a flexibilidade da máquina: uma vez que o destino é considerado um número real, ele pode ser expresso de todas as formas que o *Spectrum* reconhece para um número. Por exemplo, 1000+X, 2*A/Y, VAL«2500» seriam todos destinos válidos, desde que os respectivos resultados se situem na gama 1 a 9999. Existem alguns truques de programação inteligentes no *Basic* do *Sinclair* que dependem desta capacidade. Quando se trata de renumeração, no entanto, a representação por cinco *bytes* é um aborrecimento, tornado ainda pior pelo facto de o número de destino, tal como qualquer número real exibido numa linha programática, ser gravado, na prática, duas vezes na linha. De uma das vezes, como conjunto de caracteres para a sua observação; da segunda vez; sob a forma de cinco *bytes*.

A conclusão de tudo isto é que não só temos de renumerar a linha como também de saber se há algum GOTO ou GOSUB a apontar para a linha que está a ser renumerada. Se assim for, teremos de alterar mais dois números por métodos diferentes.

Dentro de certos limites, é isso exactamente o que este pequeno programa fará e, embora ele não seja, de modo algum, tão rápido e conveniente como um bom programa em código-máquina na obtenção dos mesmos resultados, fundido (MERGE) com os seus próprios programas fará um trabalho adequado, até que você se decida a comprar um programa comercial caro, ou a escrever o seu próprio programa.

```
9958 STOP
9959 LET T$="": LET X=23635
9960 DEF FN A(X)=PEEK X+256*PEEK
 (X+1)
9961 DEF FN B(S)=256*PEEK S+PEEK
 (S+1)
9962 LET S=FN A(X)
9963 LET LINE=FN B(S): IF LINE>=
9958 THEN GO TO 9976
9964 LET LENGTH=FN A(S+2)
9965 IF PEEK (S+4)=234 THEN GO T
O 9969
9966 FOR I=S+4 TO S+LENGTH+2
9967 IF PEEK I=236 OR PEEK I=237
 THEN GO SUB 9971
9968 NEXT I
9969 LET S=S+LENGTH+4
9970 GO TO 9963
9971 IF PEEK (I+5)=14 THEN GO TO
 9974
9972 PRINT "NON-STANDARD COMMAND
, LINE ";LINE
9973 STOP
9974 LET T$=T$+STR$ I+CHR$ PEEK
(I+1)+CHR$ PEEK (I+2)+CHR$ PEEK
(I+3)+CHR$ PEEK (I+4)
9975 RETURN
9976 LET BASE=1000: LET X=23635
```



```
9977 LET S=FN A(X)
9978 LET LINE=FN B(S): IF LINE>=
9958 THEN STOP
9979 LET LENGTH=FN A(S+2)
9980 FOR I=1 TO LEN T$ STEP 9
9981 IF VAL T$(I+5 TO I+8)=LINE
THEN GO SUB 9990
9982 NEXT I
9983 POKE S,INT (BASE/256)
9984 POKE S+1,BASE-256*INT (BASE
/256)
9985 LET BASE=BASE+10
9986 LET S=S+LENGTH+4
9987 GO TO 9978
9990 FOR J=1 TO 4
9991 POKE (VAL T$(I TO I+4)+J),C
ODE (STR$ BASE)(J)
9992 NEXT J
9993 LET BYTE1=128+INT (LN BASE/
LN 2+1)
9994 LET BYTE2=BASE*65536/(2^(BY
TE1-128))
9995 LET MEMORY=VAL T$(I TO I+4)
9996 POKE MEMORY+6,BYTE1
9997 POKE MEMORY+7,INT (BYTE2/25
6)-128
9998 POKE MEMORY+8,BYTE2-256*INT
 (BYTE2/256)
9999 RETURN
```

O programa não está escrito por módulos, visto que o objectivo é amontoar o mais possível no mínimo de números de linha possível. Assim que entender o método, poderá perfeitamente querer encurtar o programa utilizando linhas multienunciado.

Comentário

Linha 9958. Esta linha assegura que a normal execução da parte principal do programa a ser renumerado nunca atinja a rotina de renumeração.

Linha 9959. T$ será utilizada para registar cada ocorrência de um GOTO ou GOSUB no programa, conjuntamente com a respectiva localização na memória do *Spectrum*. X é o endereço, na memória do *Spectrum*, do PROG «variável de sistema», ou seja, dois *bytes* que registam o endereço de partida na memória do programa *Basic* corrente.

Linha 9960. Esta função deverá ser reconhecível como o endereço inicial do programa *Basic* corrente. Note-se que, quando a trabalhar para os seus próprios objectivos, o *Spectrum* armazena números de dois *bytes* «ao invés»: vem em primeiro lugar o menos significativo dos dois *bytes*.

Linha 9961. Apenas para o confundir, os números de linha, que também são constituidos por dois *bytes*, são armazenados ao contrário, da forma correcta. Esta função traduzirá a representação de números de linha de dois *bytes* para uma forma mais reconhecivel.

Linha 9962. S é igualada ao endereço do inicio do programa.

Linha 9963. LINE é igualada ao número de linha corrente encontrado neste ponto. Assim que a rotina tiver trabalhado no programa até ao ponto em que atinge a própria rotina de renumeração, esta parte do programa é abandonada.

Linha 9964. Os dois *bytes* a seguir ao número de linha, em qualquer das linhas, registam a extensão da linha. Este valor pode ser utilizado para avançar, com facilidade, para o princípio da linha seguinte.

Linha 9965. As linhas começadas por REM (CHR$234) não são examinadas, em busca de GOTO e GOSUB. Isto significa que, se você tiver um GOTO calculado, como 1000*X, a que o programa não pode fazer frente, bastar-lhe-á colocar um REM temporário no inicio da linha e não haverá mais problemas.

Linhas 9966-9968. A linha é examinada, *byte* a *byte*, em busca dos caracteres representativos de GOTO e GOSUB (códigos 236 e 237, respectivamente).

Linhas 9969-9970. Se nada for encontrado, a rotina passa para o inicio da linha seguinte.

Linhas 9971-9973. Se a execução atingiu este ponto, isso será porque foi encontrado um carácter na linha com o código de 236 ou 237. Estas linhas verificam se se trata de um GOTO ou GOSUB normal, que a rotina possa tratar. Isto é executado tomando por base que todos os destinos estarão visivelmente representados quando se observar a linha por quatro digitos, por exemplo, GOTO 1000, GOTO 0001. Se isto for verdadeiro, então o quinto carácter a seguir a GOTO será 14, o que indica ao *Spectrum* que se segue um número de cinco *bytes* (a segunda das duas representações do destino). Se não for este o caso, a rotina pára. Em raras ocasiões, a rotina encontrará *bytes* contendo 236 ou 237 que nada tenham a ver com GOTO ou GOSUB e parará. Após ter identificado o problema, a única solução é introduzir um REM temporário no inicio da linha ofensiva.

Linhas 9974-9975. Se a rotina descobrir um GOTO ou GOSUB padrão, armazena o endereço em T$, juntamente com os quatro *bytes* que formam o destino.

Linha 9976. Quando a rotina atinge este ponto, já passou através do programa uma vez, registando a posição de todos os GOTO ou GOSUB e as linhas para que estes apontam. O programa cria agora uma variável BASE, que será usada para o primeiro número de linha.

Linhas 9977-9978. A rotina começa de novo, no início do programa, e começa a traduzir números de linha. Assim que atingir a linha 9958, a renumeração ficará completa.

Linhas 9980-9982. Após ter estabelecido o número de linha, a rotina esquadrinha T$ para ver se se trata de uma linha apontada, em qualquer altura, por um GOTO ou GOSUB. Se assim for, passará a renumerar o GOTO ou GOSUB, da forma que será explicada mais adiante.

Linhas 9983-9987. O valor de BASE é colocado na posição ocupada pelo antigo número de linha. BASE é aumentada em 10 e a rotina passa ao número de linha seguinte.

Linhas 9990-9992. Se houver um GOTO ou GOSUB para ser renumerado, a rotina mete, em primeiro lugar, os caracteres que constituem BASE nos quatro *bytes* posteriores a GOTO ou GOSUB.

Linhas 9993-9998. Nesta secção passamos à manipulação da complexa representação por cinco *bytes* de números empregues no *Spectrum*. Esta tarefa é simplificada pelo facto de apenas estarmos a tratar com números positivos, dentro da gama 1 a 9999. Isto significa que, em qualquer circunstância, apenas teremos de tratar com os três primeiros dos cinco *bytes*. Existem duas tarefas básicas. A primeira é a de determinar o expoente do destino renumerado, quando expresso em binário — ou seja, o número de lugares antes do ponto decimal, quando o número é escrito em binário. O primeiro *byte* de qualquer número positivo armazenado pelo *Spectrum* consistirá em 128, acrescido deste expoente, valor que é criado pela linha 9993.

Os dois *bytes* seguintes registam a representação binária corrente do destino. Uma vez que já possuímos o expoente para nos dizer quantos caracteres existem neste número binário, podemos armazenar o número, com o seu primeiro «1», na primeira das oito posições do *byte* seguinte. Assim, se a representação binária fosse 1010101010, isto seria armazenado nos dois *bytes* posteriores ao *byte* expoente, visto que 1010101010000000 e o *byte* expoente registarão o facto de apenas os primeiros dez dígitos serem significativos. A nossa tarefa final é transformar o primeiro «1» do nosso número binário num zero, indicando ao *Spectrum* que este número é positivo. O «1» será reintegrado quando o número for avaliado.

O número necessário é criado pela linha 9994, sem qualquer referência óbvia a números binários. O expoente e o destino traduzido são então colocados nos primeiros três *bytes* da representação de cinco *bytes*. A rotina continua agora com a renumeração do resto do programa.

Ensaio de 5.4

A primeira regra a observar ao ensaiar uma rotina como esta é a de a gravar, antes de tentar utilizá-la. Programas que enchem a memória do *Spectrum* podem correr mal e bloquear a memória. Isto

não tem importância, quanto ao *Spectrum*, mas você poderia perder o seu programa. Após ter salvo a sua rotina de renumeração, precisará de lhe dar algo em que trabalhar; isto apenas terá de ser um programa de três linhas, do tipo 1 GOTO 0005, 2 REM, 5 REM. Modifique agora o segundo enunciado, na linha 9963, para IF LINE = 9958 THEN STOP. Corra o programa, começando em 9959. Assim que o programa parar, exiba T$. Deverá ter este aspecto: — 237590005.

Se este teste for satisfatório, corrija a linha 9963 e corra o programa novamente, a partir de 9959. Desta vez ele deverá realmente renumerar as linhas. Imprima BYTE 1 e BYTE 2, que devem ser 138 e 65280, respectivamente. Se, neste ponto, o programa não funcionar, experimente inserir alguns judiciosos enunciados STOP em vários pontos, para verificar se o procedimento acima descrito está a ser seguido.

Sumário

Se conseguir escrever uma rotina destas em código-máquina, passe à prática. Será cem vezes mais rápido. Mas não se deixe arrebatar pela ideia de que brincar com coisas como a representação de cinco *bytes* do *Sinclair* não é possível em *Basic*. O pretensiosismo no código-máquina pode resultar em programas bastante horrorosos que não levam a lado nenhum, embora o façam depressa. Por isso, seja cauteloso, se tiver mesmo de deixar o conforto do *Basic* para atingir os seus objectivos. Talvez, daqui a dez anos, a única diferença entre um programa em *Basic* e um versão em código-máquina seja apenas de um milésimo de segundo.

Avançando mais

1) Eu prefiro o meu programa numerado a partir de 1000, em partes de 10; você pode não preferir. Por que não acrescentar um mecanismo simples que permita ao utilizador especificar a base (BASE) e o espaçamento entre as linhas?

2) Veja até onde pode ir no encurtar da rotina, pela utilização de funções definidas.

3) O *Spectrum* pode tratar GOTO que apontem para uma

linha não existente. Esta rotina não renumerará tais GOTO. Acrescente ao programa uma função que esquadrinhe os números de linha, verificando se existe um destino para cada GOTO ou GOSUB.

5.5 UNIFICHA II

Com o «Unificha II», voltamos ao tema do capítulo de abertura do livro, designadamente o arquivo flexível de grandes quantidades de informações. No programa que se segue podemos observar uma nova aplicação dos métodos anteriormente discutidos no cap. 1, voltando a aperceber-nos das vantagens do nosso método modular. A renumeração do programa é, no essencial, a mesma do programa «Unificha» original e, contrariamente aos outros programas deste livro, as alterações feitas não foram renumeradas. Isto é intencional, para que você possa identificar exactamente onde foram feitas alterações e acrescentos ao programa original e para que você possa referir-se aos comentários da versão original e não só aos comentários da versão corrente.

A nova aplicação a que se destina o presente programa é a do banco de dados. Um banco (ou base) de dados, na sua expressão mais simples, é uma colecção de informações, de preferência incorporando formas de adição aos armazenamentos correntes e formas de descobrir o que lá se encontra armazenado. De modo diferente do das estruturas claras e arrumadas das fichas criadas pelo programa «Unificha» original, o banco de dados deve poder aceitar informações cuja estrutura não pode ser prevista tão facilmente. A estrutura pode variar grandemente de acesso para acesso. Espera-se, muitas vezes, que o banco de dados aplique um método mais sofisticado de busca, possibilitando ao utilizador especificar um número de características que deveriam estar presentes na busca de acesso.

Trata-se de uma especificação bastante difícil. No entanto, tal é o poder do método modular na elaboração de programas que a duração da feitura do presente programa se ficou algo abaixo das três horas, desde a ideia original até à versão acabada. Durante o que vai seguir-se, os módulos apenas são comentados se diferirem significativamente do módulo equivalente no «Unificha». Ao introduzir o programa, aconselha-se o utilizador a introduzir em primeiro lugar o programa «Unificha» original, trabalhando depois nas modificações concordantes com a listagem para o corrente programa.

MÓDULO 5.5.1

```
1000 PAPER 7: CLS : BORDER 7: IN
K 6: PAPER 0: PRINT PAPER 2;"
     UNIFILE                  "
1010 PRINT '"FUNCTIONS AVAILABLE
:"
1020 PRINT '"      1)SET UP NEW F
ILE"
1030 PRINT '"      2)ENTER INFORM
ATION"
1040 PRINT '"      3)SEARCH/DISPL
AY/CHANGE"
1045 PRINT '"      4)TYPE NAMES"
1050 PRINT '"      5)STOP"
1060 PRINT '''"PLEASE ENTER WHICH
 YOU REQUIRE."
1070 INPUT Z$
1080 CLS
1090 IF Z$="1" THEN GO SUB 1210
1100 IF Z$="2" THEN GO SUB 1440
1110 IF Z$="3" THEN GO SUB 2180
1115 IF Z$="4" THEN GO SUB 3400
1120 IF Z$="5" THEN GO SUB 1150
1130 CLS
1140 GO TO 1000
1150 PRINT AT 10,5; INK 7; PAPER
 2;"FILING SYSTEM CLOSED"
1160 BEEP 2,2
1180 INPUT "Have you input new i
nformation  you wish to save? (Y
```

```
/N)";Q$: IF Q$="N" THEN STOP
1190 SAVE "UNIFILE": PRINT '"Rew
ind tape, then Press any key to
VERIFY": PAUSE 0: VERIFY "UNIFIL
E": STOP
```

O módulo é modificado nas linhas 1045 e 1115, tendo em conta uma nova função, que será explicada adiante.

MÓDULO 5.5.2

```
1200 REM ************************
1210 REM INITIALISE
1220 REM ************************
1230 DIM A$(50,10)
1350 DIM B$(28000)
1360 LET B$(1 TO 4)=CHR$ 2+CHR$
0+CHR$ 2+CHR$ 255
1370 DEF FN A()=256*CODE Y$(2*S-
1)+CODE Y$(2*S)
1380 DEF FN A$()=B$(C TO C+CODE
B$(C)-1)
1390 LET P=5
1400 LET Y$=CHR$ 0+CHR$ 1+CHR$ 0
+CHR$ 3
1410 LET N=2
1420 RETURN
```

Foi apagada uma grande parte deste módulo, tendo em conta o facto de que não haverá qualquer estrutura regular de nomes para os assuntos, em cada acesso. Haverá, no entanto, a possibilidade de juntar nomes a assuntos, nomes esses que serão armazenados na disposição A$.

MÓDULO 5.5.3

```
2750 REM ************************
2760 REM FUNCTIONAL SUBROUTINES
2770 REM ************************
2780 INPUT Q$
2785 IF LEN Q$>1 THEN RANDOMIZE
: IF Q$(LEN Q$-1)="#" THEN LET Q
$=Q$( TO LEN Q$-1)+CHR$ (VAL Q$(
LEN Q$))
2787 IF LEN Q$>2 THEN RANDOMIZE
: IF Q$(LEN Q$-2)="#" THEN LET Q
$=Q$( TO LEN Q$-2)+CHR$ (VAL Q$(
LEN Q$-1 TO ))
2790 LET Q$=CHR$ (LEN Q$+1)+Q$
2800 RETURN
2810 PRINT A$(I,2 TO CODE A$(I,1
));":";
2820 RETURN
2850 IF FN A$()(2 TO )="*" THEN
RETURN
2860 IF FN A$()(LEN FN A$()-1)="
#" THEN PRINT A$(CODE FN A$()(LE
N FN A$()));":";
2870 PRINT FN A$()(2 TO LEN FN A
$()-2*(FN A$()(LEN FN A$()-1)="#
")
2880 LET C=C+CODE B$(C)
2890 GO TO 2850
```

É alterada a pequena rotina na linha 2780, para que possa reconhecer números de rótulo, precedidos por #, traduzindo o número seguinte para um carácter único desse código. Alterou-se a rotina da linha 2850, para que possa imprimir assuntos até detectar o delimitador «*». Os rótulos apropriados são impressos pela linha 2860, se # estiver presente.

MÓDULO 5.5.4

```
1430 REM ************************
1440 REM NORMAL INPUT
1450 REM ************************
1460 LET R$=""
1470 PRINT PAPER 2;"          EN
TRIES                "
1480 PRINT '"COMMANDS AVAILABLE:
"
1490 PRINT '">ENTER ITEM SPECIFI
ED"'">""ZZZ"" TO QUIT"
1500 PRINT "**********************
************"
1510 PRINT "FILE SIZE:";P-1;"/";
LEN B$
1540 GO SUB 2780
1550 IF Q$(LEN Q$-1)="#" THEN PR
INT A$(CODE Q$(LEN Q$));":";
1580 PRINT Q$(2 TO LEN Q$-2*(Q$(
LEN Q$-1)="#"))
1590 IF LEN Q$>=4 THEN RANDOMIZE
 : IF Q$(2 TO 4)="ZZZ" THEN RETU
RN
1600 LET R$=R$+Q$
1610 IF Q$(2)<>"*" THEN GO TO 15
20
1620 CLS
1630 GO SUB 1660
1640 GO TO 1440
```

De preferência a pedir um número especificado de assuntos, este módulo aceita agora assuntos até deparar com um assunto que consiste em «*», que tomará como indicador de fim do acesso. As linhas 1550 e 1580 referem-se ao método de rotulação do assunto. Para o conseguir, o utilizador acrescenta #, seguido de um número

entre 1 e 50 no final do assunto. Isto é tomado como indicação do número de um assunto na disposição A$, e a linha relevante de A$ é exibida pela linha 1550, se for detectado #. A linha 1580 assegura que o número não será impresso.

MÓDULO 5.5.5

```
1650 REM ************************
1660 REM PLACE DATA IN FILE
1670 REM ************************
1680 IF P+LEN R$-1<LEN B$ THEN G
O TO 1730
1690 PRINT AT 14,10;"FILE NOW FU
LL."
1700 PRINT '''   Press any key t
o continue"
1710 PAUSE 0
1720 RETURN
1730 LET POWER=INT (LN (N-1)/LN
2)
1740 LET S=2^POWER
1750 LET T$=R$(2 TO CODE R$(1))
1760 FOR K=POWER-1 TO 0 STEP -1
1770 LET C=FN A()
1780 LET U$=FN A$()(2 TO )
1790 LET S=S+(2^K)*(T$>U$)-(2^K)
*(T$<U$)
1810 IF S>N-1 THEN LET S=N-1
1820 IF S<2 THEN LET S=2
1830 NEXT K
1840 LET C=FN A()
1850 LET U$=FN A$()(2 TO )
1860 IF T$<U$ THEN LET S=S-1
1870 LET B$(P TO P+LEN R$-1)=R$
1880 LET N=N+1
1890 LET Y$=Y$(1 TO 2*S)+CHR$ IN
```

```
T (P/256)+CHR$ (P-256*INT (P/256
))+Y$(2*(S+1)-1 TO )
1900 LET P=P+LEN R$
1910 RETURN
```

Não é feita qualquer alteração aqui.

MÓDULO 5.5.6

```
2170 REM ************************
2180 REM SEARCH
2190 REM ************************
2200 LET S=2
2210 PRINT PAPER 2;"
SEARCH                "
2220 PRINT "COMMANDS AVAILABLE:"
2230 PRINT ">INPUT ITEM FOR NORM
AL SEARCH"'">PRECEDE WITH ""SSS"
" FOR"'" SPECIAL SEARCH"'">PRECE
DE WITH ""III"" TO SEARCH"'" FOR
 FIRST CHARACTER OF ENTRY"'">""E
NTER"" FOR FIRST ITEM ON FILE"
2235 PRINT ">""MMMM"" FOR MULTIP
LE SEARCH"
2240 PRINT "**********************
***********"
2250 PRINT "INPUT SEARCH ITEM:";
2260 GO SUB 2780
2270 PRINT Q$(2 TO )
2280 LET S$=Q$
2290 IF LEN S$=1 THEN GO TO 2510
2300 LET C=FN A()
2310 IF LEN S$<5 THEN GO TO 2430
2311 IF S$(2 TO 4)<>"MMM" THEN G
O TO 2320
2312 PRINT "NUMBER OF SEARCH ITE
```



```
MS? ";: INPUT SEARCH: PRINT SEAR
CH: DIM M$(SEARCH,10): FOR K=1 T
O SEARCH: PRINT "SEARCH ITEM ";K
;": ";: GO SUB 2780: PRINT Q$(2
TO ): LET M$(K)=Q$: NEXT K
2313 LET S1=S
2314 FOR K=1 TO SEARCH: LET S$="
MMMM"+M$(K,2 TO CODE M$(K,1)): G
O SUB 2940: IF C4=0 THEN RETURN
2316 IF S=S1 THEN NEXT K: GO TO
2510
2318 GO TO 2313
2320 IF S$(2 TO 4)<>"III" THEN G
O TO 2390
2330 FOR I=S TO N
2340 LET S=I
2350 LET C=FN A()
2360 IF B$(C+1)=S$(5) THEN GO TO
 2510
2370 NEXT I
2380 RETURN
2390 IF S$(2 TO 4)<>"SSS" THEN G
O TO 2430
2400 GO SUB 2920
2410 IF C4=1 THEN GO TO 2510
2420 RETURN
2430 IF FN A$()=S$ THEN GO TO 25
10
2450 IF FN A$()=CHR$ 2+CHR$ 255
THEN RETURN
2460 LET C=C+CODE B$(C)
2470 IF B$(C-1)<>"*" THEN GO TO
2430
2480 LET S=S+1
2490 LET C=FN A()
2500 GO TO 2430
2510 LET C=FN A()
2520 LET C4=0
```

```
2530 IF FN A$( )=CHR$ 2+CHR$ 255
THEN RETURN
2540 CLS
2550 PRINT "ENTRY ";S-1;"'s"
2560 GO SUB 2850
2570 LET S=S+1
2580 PRINT AT 16,0; PAPER 2;"
     SEARCH                 "
2590 PRINT "COMMANDS AVAILABLE:"
2600 PRINT ">""ENTER"" TO DISPLA
Y NEXT ITEM"'">""ZZZ"" TO QUIT F
UNCTION"'">""AAA"" TO AMEND"'">"
"CCC"" TO CONTINUE SEARCH"
2610 INPUT P$
2620 CLS
2625 IF LEN S$>=4 THEN RANDOMIZE
 : IF P$="CCC" AND S$(2 TO 4)="M
MM" THEN GO TO 2313
2630 IF P$="CCC" THEN GO TO 2300
2640 IF P$="" THEN GO TO 2510
2650 IF P$<>"AAA" THEN GO TO 271
0
2660 LET C=FN A()
2670 CLS
2680 GO SUB 1930
2710 IF P$="ZZZ" THEN RETURN
2720 IF P$="AAA" THEN RETURN
2730 CLS
2740 GO TO 2260
```

Entre a linha 2311 e 2318 é acrescentada uma pequena rotina a este módulo, a qual aceita um número de assuntos de busca, solicitando depois o módulo de busca especial para cada assunto, sucessivamente. Se todas as combinações de caracteres especificadas forem encontradas dentro de um qualquer acesso único, essa acesso será então exibido. Note-se igualmente a pequena alteração à ementa, na linha 2235, e ao comando de continuação, em 2625.

Deverá agora poder exibir assuntos, embora não possa utilizar a busca especial nem a busca múltipla. Se especificar números de rótulo, aquando da introdução, o assunto deverá ser sempre exposto inserido em dez caracteres e precedido por «:».

MÓDULO 5.5.7

```
2910 REM ************************
2920 REM SPECIAL SEARCH
2930 REM ************************
2940 LET C4=0
2950 FOR H=S TO N-1
2960 LET S=H
2970 LET C=FN A( )
2980 LET C1=C
3000 LET C1=C1+CODE B$(C1)
3010 IF B$(C1-1)<>"*" THEN GO TO
 3000
3020 FOR J=C+1 TO C1-LEN S$+5
3030 IF B$(J TO J+LEN S$-5)<>S$(
5 TO ) THEN GO TO 3060
3040 LET C4=1
3050 RETURN
3060 NEXT J
3070 NEXT H
3080 LET C4=0
3090 RETURN
```

É feita uma pequena alteração a este módulo, tendo em conta a falta de uma estrutura regular no acesso.

Ensaio do módulo 5.5.7

Deverá agora poder levar a efeito buscas de combinações de caracteres — assunto precedido por SSS. Deverá igualmente poder introduzir o modo de busca múltipla.

MÓDULO 5.5.8

```
1920 REM ************************
1930 REM CHANGE ENTRY
1940 REM ************************
1950 LET S=S-1
1960 LET C=FN A( )
1970 LET R$=""
1980 PRINT "ENTRY ";S-1;":-"
2000 IF FN A$( )(2)="*" THEN LET
R$=R$+CHR$ 2+"*": GO SUB 3130: G
O SUB 1660: RETURN
2010 PRINT FN A$( )(2 TO LEN FN A
$( )-1);
2015 IF FN A$( )(LEN FN A$( )-1)="
#" THEN PRINT CODE (FN A$( )(LEN
FN A$( )))
2017 IF FN A$( )(LEN FN A$( )-1)<>
"#" THEN PRINT FN A$( )(LEN FN A$
( ))
2020 PRINT AT 16,0; PAPER 2;"
       AMEND                  "
2030 PRINT "COMMANDS AVAILABLE:"
2040 PRINT ">""ENTER"" LEAVES IT
EM UNCHANGED"'">""ZZZ"" DELETES
WHOLE ENTRY"'">CHANGED ITEM"'">N
EW ITEM ENDING WITH ""*"""
2050 GO SUB 2780
2060 IF LEN Q$=1 OR Q$(LEN Q$)="
*" THEN LET R$=R$+FN A$( )
2070 LET C=C+CODE B$(C)
```



```
2080 CLS
2090 IF LEN Q$=1 THEN GO TO 2000
2100 IF Q$(2 TO )="ZZZ" THEN GO
SUB 3130: RETURN
2105 IF Q$(LEN Q$)="*" THEN LET
Q$=Q$(2 TO LEN Q$-1): GO SUB 278
5
2110 LET R$=R$+Q$
2120 GO TO 2000
2130 GO SUB 3130
2140 IF Q$(2 TO )="ZZZ" THEN RET
URN
2150 GO SUB 1660
2160 RETURN
```

As linhas 2015 e 2017 reflectem a necessidade de traduzir um número de rótulo, se presente, do carácter do código relevante para um número corrente. Se não houver número de rótulo, serão impressos os dois últimos caracteres do assunto. A ementa e as linhas subsequentes são modificadas, levando em conta os assuntos com delimitador «*». Sempre que um destes for introduzido, será armazenado no acesso, a seguir ao assunto a ser exibido correntemente no topo do *écran*. Isto significa que não pode ser inserido um novo assunto no princípio de um acesso.

MÓDULO 5.5.9

```
3100 REM ************************
3110 REM TELESCOPE FILE
3120 REM ************************
3130 LET C=FN A( )
3140 LET SHIFT=1000
3150 LET C1=C
3160 LET C3=C
```

```
3180 LET C1=C1+CODE B$(C1)
3190 IF B$(C1-1)<>"*" THEN GO TO
 3180
3200 LET C2=C1-C
3210 FOR I=C1 TO LEN B$-1 STEP S
HIFT
3220 IF LEN B$-I+1<SHIFT THEN LE
T SHIFT=LEN B$-I+1
3230 LET S$=B$(I TO I+SHIFT-1)
3240 LET B$(C TO C+SHIFT-1)=S$
3250 LET C=C+SHIFT
3260 NEXT I
3270 LET Y$=Y$(1 TO 2*(S-1))+Y$(
2*(S+1)-1 TO )
3280 FOR I=1 TO N-1
3290 LET S=I
3300 LET C=FN A()
3310 IF C<=C3 THEN GO TO 3350
3320 LET C=C-C2
3330 LET Y$(2*I-1)=CHR$ INT (C/2
56)
3340 LET Y$(2*I)=CHR$ (C-256*INT
 (C/256))
3350 NEXT I
3360 LET P=P-C2
3370 LET N=N-1
3380 RETURN
```

São feitas alterações na linha 3190, por apagamento da linha 3170.

Ensaio do módulo 5.5.9

A função de ementa deverá agora estar completamente operacional, permitindo apagamentos, alterações e inserções.


```
3400 REM ************************
3410 REM ITEM TYPES
3420 REM ************************
3430 FOR I=1 TO 50 STEP 10
3440 CLS : FOR J=I TO I+9
3450 PRINT J;")";A$(J)
3460 NEXT J
3470 PRINT '''"COMMANDS:"
3480 PRINT '">ZZZ=QUIT"
3490 PRINT ">III=ITEM"
3495 PRINT ">NNN=NEXT PAGE"
3500 INPUT Q$: IF Q$="ZZZ" THEN
RETURN
3505 IF Q$="NNN" THEN CLS : NEXT
 I: RETURN
3510 IF Q$="III" THEN INPUT "NUM
BER? ";TYPE: INPUT "TYPE NAME? "
;Q$: LET A$(TYPE)=Q$: CLS : GO T
O 3440
3520 GO TO 3500
```

Este curto módulo permite ao utilizador introduzir rótulos para assuntos em posições especificadas, na disposição A$. Não há qualquer provisão especifica para apagamentos, o que, mesmo assim, pode ser feito simplesmente especificando a linha relevante e introduzindo uma sequência vazia como novo conteúdo.

Ensaio do módulo 5.5.10

Se especificou previamente números de rótulo, para alguns dos assuntos, deverá agora poder fornecer algum conteúdo para os rótulos, vendo os rótulos exibidos em frente aos assuntos relevantes pelo módulo competente.

Sumário

Seria insensato sugerir que um programa como este é tão útil como um banco de dados em rápido código-máquina, capaz de realizar uma busca complexa em segundos. Ainda assim, o programa funciona e tem capacidades superiores a alguns programas baratos de banco de dados, vendidos para os microcomputadores pessoais. A busca múltipla é um luxo consumidor de tempo, que não deverá querer utilizar com frequência numa ficha vasta. Poderá, devido às suas necessidades específicas, vir a utilizar esta versão do «Unificha» mais assiduamente que o original.

A verdadeira lição a tirar é a de que o método modular tornou simples uma vasta e principal aplicação, o que permitiu identificar claramente as áreas necessitadas de modificação.

Avançando mais

1) Tendo sido avisados contra o pretensiosismo em código-máquina, tem de se admitir que, com uma aplicação destas, estamos no limiar do que pode, com utilidade, ser alcançado em *Basic*, com uma máquina com a velocidade do *Spectrum*. As minhas próprias versões do «Unificha» para o ZX81 incorporam curtas rotinas em código-máquina, na realização de buscas consumidoras de tempo. Se as aplicações do banco de dados vão ser cruciais para a sua utilização do *Spectrum*, então essa será a direcção a seguir.

5.6 DACTILÓGRAFO

Nem todos os programas úteis devem ter centenas de linhas de extensão. Este dar-vos-á um grande empurrão no sentido da escrita com um mínimo de toques, se usado correctamente. É um programa curto porque, tal como acontece com o explicador de geografia anteriormente observado, é deixada ao utilizador a liberdade de introduzir e apagar o material em que o programa irá trabalhar, sob a forma de enunciados DATA. A partir daqui não há longos módulos para aceitar a introdução de dados e apagar assuntos de dados. O programa é uma advertência da constante necessidade de examinar aquilo que você pretende fazer com o *Spectrum*, fazendo a pergunta: tem de ser tão sofisticado? Por vezes a resposta é sim, sem qualquer dúvida, tal como no caso do verdadeiro programa de utilidade múltipla. Frequentemente, é óbvio que a estrutura de um programa é tal que é muito pouco provável que venha a ter alguma utilidade, a não ser no objectivo único para que foi desenhado.

MÓDULO 5.6.1

```
1270 REM **********************
1280 REM USR CHARACTERS
1290 REM **********************
1300 RESTORE : FOR I=0 TO 63: RE
AD BYTE: POKE USR "A"+I,BYTE: NE
XT I: RETURN
1310 DATA 0,0,0,0,0,7,4,4
1320 DATA 0,0,0,0,0,255,0,0
1330 DATA 0,0,0,0,0,224,32,32
1340 DATA 4,4,4,4,4,4,4,4
1350 DATA 32,32,32,32,32,32,32,3
2
1360 DATA 4,4,7,0,0,0,0,0
1370 DATA 0,0,255,0,0,0,0,0
1380 DATA 32,32,224,0,0,0,0,0
1390 RETURN
```

Este módulo preenche os primeiros oito caracteres da área de gráficos definidos pelo utilizador com os caracteres necessários para desenhar um pequeno quadrado em volta de uma letra. Estes quadrados serão utilizados para representar teclas de máquina de escrever no *écran*.

MÓDULO 5.6.2

```
1000 REM ************************
1010 REM PRINT KEYBOARD
1020 REM ************************
1030 INK 0: PAPER 6: CLS : DIM O
$(32): GO SUB 1270
1040 CLS : PRINT "ABCABCABCABCAB
CABCABCABCABCABC  D ED ED ED ED
ED ED ED ED E  FGHFGHFGHFGHFG
HFGHFGHFGHFGHFGH"
1050 PRINT " ABCABCABCABCABCABCA
BCABCABCABC  D ED ED ED ED ED ED
 ED ED ED E  FGHFGHFGHFGHFGHFGHF
GHFGHFGHFGH"
1060 PRINT "  ABCABCABCABCABCABC
ABCABCABCABC  D ED ED ED ED E
D ED ED ED E  FGHFGHFGHFGHFGHFGH
FGHFGHFGHFGH"
1070 PRINT " ABCABCABCABCABCABCA
BCABCABCABC  D ED ED ED ED ED ED
 ED ED ED E  FGHFGHFGHFGHFGHFGHF
GHFGHFGHFGH"
1080 PRINT OVER 1;AT 1,0;" 1  2
 3  4  5  6  7  8  9  0"
1090 PRINT OVER 1;AT 4,0;"  Q  W
  E  R  T  Y  U  I  O  P"
1100 PRINT OVER 1;AT 7,0;"   A
S  D  F  G  H  J  K  L  ENTER"
1110 PRINT OVER 1;AT 10,0;"  CAPS
Z  X  C  V  B  N  M  SYMBOL"
```

Este módulo desenha uma cópia aproximada do teclado de uma máquina de escrever, na metade superior do *écran*, utilizando os caracteres definidos pelo módulo anterior.

Comentário

Linhas 1040-1070. Estas sequências de letras são introduzidas no modo de gráficos e criarão, de facto, o contorno quadrangular das teclas.

Linhas 1080-1110. Cada tecla é rotulada com a letra principal, ou número, se o contiver. A tecla de espaço é deixada em branco e as teclas CAPS SHIFT, SYMBOL SHIFT e ENTER são rotuladas com as suas iniciais invertidas.

Ensaio do módulo 5.6.2

Coloque um STOP temporário no final do módulo e deverá poder imprimir um teclado reconhecível no *écran*.

MÓDULO 5.6.3

```
1400 REM ************************
1410 REM DATA FOR TESTS
1420 REM ************************
1430 DATA "THIS IS A SPECTRUM TY
PING TEST"
1440 DATA "JUST TYPE WHAT YOU SE
E"
1450 DATA "DONT LOOK AT THE KEYB
OARD"
1460 DATA "STOP"
```

Este módulo será utilizado para conter os dados dos ensaios de escrita. As sequências individuais não deverão ser maiores do que 32 caracteres. O módulo deverá ser finalizado com uma linha DATA contendo STOP, o qual é usado como um sinal para recolocar o indicador DATA, no módulo seguinte.

```
1120 REM **********************
1130 REM ACCEPT INPUT
1140 REM **********************
1150 LET TOTAL=0: LET RIGHT=0
1160 READ A$: IF A$="STOP" THEN
RESTORE 1430: READ A$
1170 PRINT INK 1;AT 16,0;0$;AT 1
6,0;A$'0$: PRINT AT 17,0;
1180 FOR I=1 TO LEN A$
1190 LET T$=INKEY$: IF T$<>"" TH
EN GO TO 1210
1200 GO TO 1190
1210 LET TOTAL=TOTAL+1: BEEP .1,
30: IF T$<>A$(I) THEN PRINT PAPE
R 5;T$;CHR$ 8;: GO TO 1190
1220 LET RIGHT=RIGHT+1
1230 PRINT BRIGHT 1;T$;
1240 NEXT I: PRINT AT 21,25;INT
(RIGHT/TOTAL*100);"%  "
1250 INPUT "MORE? (Y/N) ";Q$: IF
 Q$<>"N" THEN GO TO 1160
1260 STOP
```

Este módulo imprime no *écran* uma sequência a ser escrita e convida o utilizador a copiá-la, no *écran*, sem referência ao teclado corrente. As teclas premidas incorrectamente não serão aceites, embora seja mantido um registo da taxa de erro, o qual será exibido no final de cada sequência.

Comentário

Linha 1160. Esta linha circula pelos dados, no módulo 3, e reintegra o início dos dados (DATA), caso encontre um STOP.

Linhas 1180-1240. Este desvio aceita a introdução corrente de dados — a variável do desvio apenas será incrementada se forem premidas teclas correctas.

Linha 1210. Teclas incorrectamente premidas são exibidas, para que o utilizador as veja, distinguindo-se por um fundo turquesa. Note-se o uso do carácter de controlo, CHR$ 8, para recuar a posição de impressão um lugar, logo que seja exibido o carácter incorrecto, de forma a que a posição de impressão permaneça no mesmo lugar, sempre que for introduzido um carácter incorrecto.

Linha 1230. As teclas premidas correctamente são acrescentadas à sequência que estiver a ser escrita. A característica BRIGHT é preparada para que o utilizador possa ficar certo de que foram introduzidos espaços.

Ensaio do módulo 5.6.4

Se introduziu alguns dados no módulo 3, deverá agora poder correr o programa e ver o teclado exibido. A primeira sequência da área dos dados deverá aparecer logo abaixo e o utilizador deverá poder introduzir uma tentativa de a dactilografar de novo.

Sumário

A utilidade deste programa depende de ser, ou não, tomado a sério. Utilizado como um tutor de escrita decente, tirado da biblioteca, que proporcione o género correcto de exercícios, o programa pode ser uma ferramenta muito eficiente. Para começar, você deverá achar mais fácil manter-se nas letras maiúsculas, ou minúsculas, sem as misturar com pontuação ou símbolos. Quando você estiver preparado, o programa tratará dos caracteres misturados mais os símbolos, com SYMBOL SHIFT.

Avançando mais

1) Utilizando o método salientado na p. 131 do manual do *Spectrum*, acrescente uma função de cronometragem aos dados introduzidos, alterando depois o programa para que ele imprima

uma série de sequências até, digamos, mil caracteres. Verifique o tempo quando começar a introduzir e quando o teste estiver completo, e terá, em conjunto com a percentagem de teclas premidas, uma boa indicação do seu progresso.

2) Altere a exibição, de modo a que o *écran* apenas mostre o teclado quando você cometer um erro — talvez mesmo mais de um erro no mesmo carácter — e o retire sempre que o carácter for correctamente introduzido.

CAPÍTULO 6

FINALMENTE ALGUM DIVERTIMENTO — IDEIAS CASEIRAS PARA O «SPECTRUM»

Penso que já deve ter percebido, pela apresentação global deste livro, que eu não considero os jogos como a essência e o objectivo do computador em casa. Receio bem que os jogos sejam um retrocesso para aqueles que já descobriram o fascínio da computadorização mas ainda não exploraram o modo como o poder do micro pode melhorar a vida quotidiana. Ainda assim, os jogos têm o seu lugar, dependendo, é claro, do jogo. Colocar um jogo pouco interessante num micro não o torna melhor e os caracteres definidos pelo utilizador não alteram o facto de que, em *Basic*, o *Spectrum* é demasiado lento para a prática dos jogos do tipo *Invaders* com um grau real de satisfação.

Os micros distinguem-se em jogos que requeiram pensamento. No que vai seguir-se encontrarão dois jogos que dependem muito mais do pensamento do que da rapidez de reacções. O primeiro deles, *Míssil*, parece, à primeira vista, uma cópia directa do tipo de jogo de ceifar inimigos, mas requer verdadeiros cálculos, da parte do jogador. A seguir vem o *Perseguidor*, um jogo enfurecedor em que se caça uma presa invisível. Finalmente inclui uma rotina simples de selecção, que deverá provar-se útil em tendências para *puzzles* de palavras e outros jogos que tenham de utilizar selecção sequencial ou numérica.

6.1 MÍSSIL

Não pode ganhar este jogo. No final, as vorazes hordas inimigas varrerão o *écran* e apanhá-lo-ão; a sua missão é levar consigo quantos for capaz. As suas armas consistem em misseis capazes de levar ogivas de potência variável, disparados a partir de bases de misseis. As naves inimigas, representadas por números entre 0 e 9, descem ao acaso, uma linha de cada vez, desde o topo do *écran*. Se atingirem a base do *écran*, a potência delas é abatida no seu *stock* de ogivas. Em cada etapa, o jogador é convidado a especificar a base de onde irá disparar, o ângulo tomado pelo míssil... desde a vertical até 45 graus para a direita, a altura a que o míssil explodirá, desde que não bata em nada primeiro, e a potência da ogiva — introduzida como potência de dois, até um máximo de quatro; por exemplo: 16.

Quando o missil explode, se realmente atingiu uma nave inimiga, reduzirá no dobro do seu próprio valor o poder dessa nave. Uma nave inimiga que bordeje o quadrado de explosão do missil será reduzida no valor equivalente ao da potência do missil. As naves com um espaço de um quadrado entre elas e o missil perderão poder equivalente a metade da potência do missil, e assim por diante. Se uma nave inimiga for reduzida em poder abaixo de zero desaparecerá. Por cada jogada são gerados mais inimigos.

Por cada cinco jogadas, uma nave de abastecimento, representada por «*», inicia uma descida desde o topo do *écran*, por entre as naves inimigas. Se conseguir aterrar, aumentará para 100 o seu *stock* de ogivas. Mas se, em qualquer ponto, for apanhada pelo círculo de destruição criado por um míssil, tudo ficará perdido. A pontuação consiste na quantidade total de estragos que o jogador infligir às naves inimigas e o jogo termina quando se esgotar o *stock* de ogivas. Uma complicação final é que o jogador tem uma quantidade de tempo limitada em que deve dar entrada a quaisquer dados; podem estabelecer-se vários graus de dificuldade para isto. Se o tempo for excedido ou feita uma introdução de dados incorrecta, os inimigos descem em linha sem qualquer impedimento.

MÓDULO 6.1.1

```
1000 REM ************************
1010 REM VARIABLES
1020 REM ************************
1030 DIM A$(608): DIM C$(4,6): D
IM C(4): DIM D(4,2): DIM O$(32)
1040 LET C$(1)="BASE?": LET C$(2
)="ANGLE?": LET C$(3)="DELAY?":
LET C$(4)="POWER"
1050 LET U$="": LET SCORE=0: LET
 W=100: LET S=1: LET G=1
1060 LET D(1,1)=49: LET D(1,2)=5
4: LET D(2,1)=48: LET D(2,2)=57:
 LET D(3,1)=48: LET D(3,2)=57: L
ET D(4,1)=48: LET D(4,2)=54
1070 BORDER 0: PRINT INK 2; PAPE
R 6;AT 10,8;"MISSILE ATTACK"
1080 INPUT "Please input desired
 difficulty level (0-10):";H: LE
T H=500-45*H
```

A utilização destas variáveis será explicada no decurso do programa. O módulo aceita igualmente o grau de dificuldade.

MÓDULO 6.1.2

```
1910 REM ************************
1920 REM GENERATE ALIENS
1930 REM ************************
1940 LET G=G+1: FOR I=1 TO INT (
RND*(G/5)+1)
1950 LET A=(RND*30+1)
1960 LET SH=INT (RND*10)
1970 LET A$(A)=CHR$ (48+SH)
1980 NEXT I
1990 IF INT ((G-5)/10)=(G-5)/10
THEN LET A$(A)="*"
2000 PRINT AT 0,0;A$: RETURN
```

Este módulo cria as naves inimigas.

Comentário

Linha 1940. O número de inimigos criados aumenta com o número de jogadas até ao momento.

Linha 1950. A posição de uma nave inimiga.

Linhas 1960-1970. É criado um carácter entre 0 e 9.

Linha 1990. É criada uma nave de abastecimento, por cada cinco jogadas.

Linha 2000. É impressa a disposição que detêm o quadro de jogo.

Ensaio do módulo 6.1.2

Ao solicitar este módulo deverão aparecer no *écran* números casuais.

MÓDULO 6.1.3

```
2010 REM ************************
2020 REM MOVE SCREEN
2030 REM ************************
2040 LET A$(33 TO 608)=A$(1 TO 5
76): LET A$(1 TO 32)=0$: PRINT A
T 0,0;A$
```

O quadro de jogo é deslocado para baixo, no *écran*, uma linha.

Ensaio do módulo 6.1.3

Se solicitar este módulo após o anterior, os números casuais deverão descer no *écran*.

MÓDULO 6.1.4

```
1790 REM ************************
1800 REM MISSILE TRACK
1810 REM ************************
1820 LET M=577+5*(C(1)-1)
1830 FOR I=1 TO 18
1840 IF A$(INT M)<>" " THEN RETU
RN
1850 LET A$(INT M)="+"
1860 PRINT AT 0,0;A$
1870 IF I=C(3) THEN RETURN
1880 LET A$(INT M)=" "
1890 LET M=M-32+SIN C(2)
1900 NEXT I: RETURN
```

Este módulo movimenta um « + », que representa a subida do míssil do jogador no *écran*.

Comentário

Linha 1820. É especificado o quadrado acima da base.

Linhas 1830-1900. Desde que não haja qualquer obstáculo e que não tenha sido obtida a altura máxima especificada, o míssil subirá 32 espaços na disposição unidimensional, voltando depois atrás alguns espaços, para representar o ângulo da direcção tomada pelo míssil. O efeito resultante é que o míssil sobe, para a direita.

Ensaio do módulo 6.1.4

Se introduzir um número, em radianos, para o ângulo do missil (C(2)), altura máxima (C(3)) e número de uma base (C(1)), então, solicite este módulo e verá o missil subir no *écran*.

MÓDULO 6.1.5

```
1450 REM **********************
1460 REM COMMAND CYCLE
1470 REM **********************
1480 INK 0: PRINT AT 20,0;O$;O$
1490 PRINT AT 21,0;"SCORE:";SCOR
E;"WARHEADS:";W;"  "
1500 PRINT AT 20,0;
1510 FOR I=1 TO 4
1520 PRINT C$(I);
1530 FOR J=1 TO H+10
1540 LET T$=INKEY$: IF T$<>"" TH
EN GO TO 1570
1550 NEXT J
1560 PRINT AT 10,5;"DEFAULT ON "
;C$(I)
1565 FOR H=1 TO 200: NEXT H: LET
 U$="": GO SUB 2010: GO SUB 1910
: GO TO 1480
1570 BEEP .1,20: IF U$<>"" OR I<
>3 THEN GO TO 1600
1580 IF CODE T$>57 OR CODE T$<48
 THEN GO TO 1560
1590 LET U$=T$: PAUSE 25: GO TO
1530
1600 IF U$="" THEN GO TO 1630
1610 IF CODE T$>57 OR CODE T$<48
 THEN GO TO 1560
1620 LET T$=U$+T$: LET U$=""
1630 IF CODE T$>57 OR CODE T$<48
 THEN GO TO 1560
1640 IF CODE T$<D(I,1) OR CODE T
$>D(I,2) THEN GO TO 1560
1650 LET C(I)=VAL T$
1660 PRINT C(I);"_" AND I<>4;
1670 NEXT I
1680 LET I=I-1
1690 LET C(2)=C(2)*PI/36
1700 LET C(4)=2^C(4)
1710 IF C(4)>W OR C(4)>16 THEN G
O TO 1560
1720 LET W=W-C(4)
1730 IF W>0 THEN GO TO 1770
1740 PRINT AT 10,3;"YOU ARE OUT
OF WARHEADS."
1750 PRINT AT 11,6;"YOUR SCORE W
AS ";SCORE
1760 STOP
1770 PRINT AT 21,0;"SCORE:";SCOR
E;"WARHEADS:";W;"  "
1780 RETURN
```

Este módulo aceita as ordens do jogador e não as cumpre se não forem mantidos os tempos correctos.

Comentário

Linhas 1510-1565. As quatro ordens, cujos nomes ficam armazenados em **C$**, são impressas e o desvio da linha 1530 concede então ao jogador um curto período para premir uma tecla. Se não for premida qualquer tecla, o programa não cumpre e os inimigos descem.

Linhas 1570-1670. O módulo verifica se uma tecla premida se inclui dentro de limites armazenados na disposição D, dependendo de qual a ordem introduzida. Se forem excedidos os limites para qualquer ordem, o módulo não cumprirá.

Linha 1690. O ângulo introduzido é multiplicado por cinco e expresso em radianos.

Linha 1710. É colocado um limite máximo de 16 na potência de um missil, visto que, para valores mais elevados, é possivel que sejam criados caracteres inválidos em certos pontos, durante o programa.

Ensaio do módulo 6.1.5

Deverá agora poder introduzir as quatro ordens, dentro dos limites estabelecidos na disposição D — linha 1060. O programa não cumprirá, caso os limites sejam excedidos ou não haja qualquer entrada de dados.

MÓDULO 6.1.6

```
1180 REM ************************
1190 REM LETHALITY
1200 REM ************************
1210 PRINT AT 20,0;0$;0$
1220 PRINT AT 20,5;"*CALCULATING
 LETHALITY*"
1230 LET M1=INT (M/32)
1240 LET M2=INT (M-32*M1)
1250 LET R=INT (.5+LN C(4)/LN 2)
1260 FOR J=M1-R TO M1+R
1270 PRINT BRIGHT 8;AT 0,0;A$
1280 PRINT BRIGHT 1;AT M1,M2-1;A
$(INT M)
1290 FOR K=M2-R TO M2+R
1300 IF 32*J+K<1 OR 32*J+K>608 T
HEN GO TO 1400
1310 IF J>=0 AND J<=21 AND K>0 A
ND K<33 THEN PRINT BRIGHT 1; OVE
R 1;AT J,K-1;" "
```



```
1320 IF A$(32*J+K)="+" THEN GO T
O 1400
1330 IF A$(32*J+K)=" " THEN GO T
O 1400
1340 LET D=ABS (K-M2)
1350 IF ABS (J-M1)>ABS (K-M2) TH
EN LET D=ABS (J-M1)
1360 LET E=INT (C(4)/(2^(D-1)))
1370 LET A$(32*J+K)=CHR$ (CODE A
$(32*J+K)-E)
1380 IF CODE A$(32*J+K)<48 THEN
LET SCORE=SCORE+CODE A$(32*J+K)-
48: LET A$(32*J+K)=" "
1390 LET SCORE=SCORE+E
1400 NEXT K
1410 NEXT J
1420 IF A$(INT M)="+" THEN LET A
$(INT M)=" "
1430 INK 0: PAPER 5: PRINT AT 0,
0;A$
1440 RETURN
```

Este módulo calcula o efeito de uma explosão nas naves inimigas.

Comentário

Linhas 1230-1240. As coordenadas dos misseis são expressas em dois termos bidimensionais.

Linha 1250. R é o alcance do círculo de destruição criado pela explosão.

Linhas 1260-1410. Com estes desvios são examinadas posições de caracteres dentro de um alcance de R, a partir do ponto de explosão do missil. Se a posição sob exame estiver vazia, ou contiver o próprio missil, não será tomada qualquer iniciativa. Se a posição contiver uma das naves inimigas, o seu valor será decrescido segun-

do a potência do missil e a distância a que este se encontra da nave inimiga. Se o valor da nave tiver sido reduzido para menos que 0, a nave será removida.

Ensaio do módulo 6.1.6

A solicitação deste módulo após o módulo 4 deverá resultar na intensificação luminosa do quadrado impresso no *écran*, na última posição do missil. A dimensão do quadrado está dependente da potência do missil.

MÓDULO 6.1.7

```
2050 REM ************************
2060 REM PENALTIES AND BONUSES
2070 REM ************************
2080 IF A$(577 TO 608)=0$ THEN R
ETURN
2090 FOR I=0 TO 31
2100 IF A$(577+I)="*" THEN LET W
=W+100
2110 IF CODE A$(577+I)>>42 THEN L
ET W=W-CODE A$(577+I)+48
2120 NEXT I
2130 LET A$(577 TO 608)=0$: PRIN
T AT 18,0;0$
2140 PRINT AT 0,0;0$
2150 IF W>0 THEN RETURN
2160 PRINT AT 10,3;"ALIEN ATTACK
 PRESSED HOME."
2170 PRINT AT 11,9;"YOU ARE DEAD
."
2180 PRINT AT 12,7;"YOUR SCORE W
AS ";SCORE
2190 STOP
```


Na eventualidade de ocupantes inimigos atingirem o fundo do *écran*, este módulo subtrai o valor deles ao *stock* de ogivas do jogador.

MÓDULO 6.1.8

```
1090 REM ************************
1100 REM MAIN LOOP
1110 REM ************************
1120 PRINT AT 19,0;"1####2####3#
###4####5####6######"
1130 GO SUB 1910
1140 INK 0: PAPER 5: PRINT AT 0,
0;A$
1150 PRINT AT 20,0;0$;0$
1170 GO SUB 1470: GO SUB 1820: G
O SUB 1180: GO SUB 2010: GO SUB
1910: GO TO 1150
```

Este módulo imprime as bases ao fundo do *écran*, solicitando depois as várias sub-rotinas em sequência.

Ensaio do módulo 6.1.8

Deverá agora poder jogar o jogo.

Sumário

Não se trata de um jogo rápido, embora valha a pena jogá-lo, pela simples razão de que é extremamente apaixonante. Jogado nos graus mais altos de dificuldade, submete o jogador a uma pressão constante para tomar decisões e, em qualquer nivel, uma pontuação respeitável (1500 e mais) requer pensamento rápido e alguns cálculos precisos.

O programa fornece igualmente um exemplo do modo como uma disposição unidimensional pode ser utilizada para armazenar um jogo a todo o *écran*, manipulando-o com facilidade. O método de dar uma introdução de dados controlada no tempo, nas linhas 1530-1550, será achado útil numa variedade de jogos onde é desejável colocar o jogador sob alguma pressão.

Avançando mais

1) Trata-se de um programa que apenas pede que o ornamentem com alguns gráficos definidos pelo utilizador. Podem definir-se caracteres para o missil e para as várias naves inimigas. Nesse caso, as linhas 1370 e 1380 terão de ser alteradas, de forma a que os caracteres procedem de uma base CODE 144, em vez de 48.

6.2 PERSEGUIDOR

Trata-se de um jogo enfurecedor, que o fará duvidar das suas capacidades mentais ou do normal funcionamento do seu *Spectrum*. É um jogo praticado num quadro de 30 por 18, no qual um dos quadrados esconde a sua presa invisível. A sua tarefa é perseguir a presa mas, de cada vez que você tenta adivinhar, a presa faz uma jogada secreta. As únicas vantagens do jogador são que a jogada da presa é sempre a mesma, em qualquer jogo, e que você tem um indicador especial que mostra a direcção da presa, de cada vez que você tenta adivinhar a posição dela. Instruções completas encontram-se impressas no próprio jogo.

Quanto às tácticas, é aconselhável tratar as duas componentes da jogada da presa — vertical e horizontal — separadamente, tentando identificar uma antes de se deixar distrair pela outra. Precisa de ser muito cuidadoso quanto à possibilidade de ter determinado a direcção correcta em que a presa se desloca. É tudo demasiado fácil quando se trabalha sob o princípio de que a presa se desloca no quadro da esquerda para a direita — princípio baseado em dois ou três palpites e na pista acompanhante — quando, durante todo o tempo, ela se deslocou na direcção oposta.


```
2180 REM ************************
2190 REM USR CHARACTERS
2200 REM ************************
2210 RESTORE : FOR I=1 TO 8: FOR
 J=0 TO 7: READ B: POKE USR CHR$
 (143+I)+J,B: NEXT J: NEXT I: RE
TURN
2220 DATA BIN 00000000,BIN 01111
000,BIN 01100000,BIN 01010000,BI
N 01001000,BIN 00000100,BIN 0000
0010,BIN 00000001
2230 DATA BIN 00001000,BIN 00011
100,BIN 00101010,BIN 01001001,BI
N 00001000,BIN 00001000,BIN 0000
1000,BIN 00001000
2240 DATA BIN 00000000,BIN 00011
110,BIN 00000110,BIN 00001010,BI
N 00010010,BIN 00100010,BIN 0100
0000,BIN 10000000
2250 DATA BIN 00010000,BIN 00100
000,BIN 01000000,BIN 11111111,BI
N 01000000,BIN 00100000,BIN 0001
0000,BIN 00000000
2260 DATA BIN 00001000,BIN 00000
100,BIN 00000010,BIN 11111111,BI
N 00000010,BIN 00000100,BIN 0000
1000,BIN 00000000
2270 DATA BIN 00000001,BIN 00000
010,BIN 00000100,BIN 01001000,BI
N 01010000,BIN 01100000,BIN 0111
1000,BIN 00000000
2280 DATA BIN 00001000,BIN 00001
000,BIN 00001000,BIN 00001000,BI
N 01001001,BIN 00101010,BIN 0001
```

```
1100,BIN 00001000
2290 DATA BIN 10000000,BIN 01000
000,BIN 00100000,BIN 00010010,BI
N 00001010,BIN 00000110,BIN 0000
1111,BIN 00000000
```

Este módulo dota os primeiros oito caracteres definidos pelo utilizador com nove setas, as quais serão usadas para indicar a direcção da presa.

Ensaio do módulo 6.2.1

A passagem deste módulo deverá resultar no preenchimento dos espaços A a H, na área definida pelo utilizador, com as setas.

MÓDULO 6.2.2

```
1900 REM **********************
1910 REM INSTRUCTIONS
1920 REM **********************
1930 PRINT INK 3;AT 0,11;"INSTRU
CTIONS"
1940 PRINT "This is a hunting ga
me."'"The hunting ground is an
18 by"'"30 board."'"The quarry
is invisible."
1950 PRINT '"Each turn the quarr
y makes a"'"secret move. This mo
ve does not"'"change during a pa
rticular hunt."
1960 PRINT '"The move can be up
to six spaces"'"up or down and s
ix spaces left"'"or right."
1970 PRINT '"Press any key for m
ore"'"instructions.": PAUSE 0: C
LS
```



```
1980 PRINT '''"Each turn consist
s of:"
1990 PRINT '"1) An invitation to
 input your"'"estimate of the po
sition of the"'"quarry."
2000 PRINT '"2) An arrow will ap
pear in your"'"specified square,
indicating the"'"direction of th
e quarry."
2010 PRINT '"3) The quarry will
move."
2020 PRINT '"Press any key to co
ntinue.": PAUSE 0: CLS
2030 PRINT '''"At the start of ea
ch turn you"'"have the opportuni
ty to review"'"the hunt so far."
2040 PRINT '"This is done by ent
ering 0 when"'"the DOWN co-ordin
ate is called"'"for."
2050 PRINT '"You can start the r
eview at any"'"previous move but
 you are"'"limited to reviewing
 20 moves"'"in any one hunt."
2060 PRINT '"These 20 reviews ma
y be taken"'"all at once or in b
atches."
2070 PRINT '"Any key to start.":
 PAUSE 0: RETURN
```

Um conjunto completo de instruções sobre o jogo. Se vai ou não incluí-las neste jogo ou noutros que desenhar, isso dependerá de quem vai jogar.

MÓDULO 6.2.3

```
2080 REM ************************
2090 REM SET DIFFICULTY
2100 REM ************************
2110 CLS
2120 PRINT '''"There is a diffic
ulty factor"'"built into the gam
e."
2130 PRINT '"This consists of an
 entirely"'"random move every so
 often."
2140 PRINT '"The difficulty fact
or ranges"'"from 0 to 10."
2150 PRINT '"0 means no random m
ove at all."
2160 INPUT INK 6;"Please input y
our desired"'"difficulty factor:
";E
2170 LET E=(11-E)*2+2+100*(E=0):
 CLS : RETURN
```

Este módulo organiza um factor de dificuldade. No caso da inserção de zero, a jogada casual apenas será feita após cem jogadas... altura em que acabará o jogo.

MÓDULO 6.2.4

```
1000 GO SUB 2190: BORDER 0: INK
1: PAPER 7: PRINT AT 10,12;"TRAC
KER"
1010 INPUT "Do yo want instructi
ons? (Y/N)"'Q$: CLS : IF Q$="Y"
OR Q$="y" THEN GO SUB 1910
1020 GO SUB 2060
1030 REM ************************
```



```
1040 REM VARIABLES
1050 REM ************************
1060 LET A=0: LET C=20: LET C1=0
: LET T=0: DIM M(100,4): DIM O$(
32)
1070 LET R1=6-INT (RND*13)
1080 LET R2=6-INT (RND*13)
1090 LET P1=INT (RND*18+1)
1100 LET P2=INT (RND*30+1)
```

Este módulo apresenta a opção das instruções e organiza as várias disposições em que o programa se baseia.

MÓDULO 6.2.5

```
1110 REM ************************
1120 REM INITIAL BOARD
1130 REM ************************
1140 CLS : LET A$="12345678901234
5678901234567890": PRINT " ";A$
1150 FOR I=1 TO 18: PRINT A$(I);
 NEXT I: GO SUB 1370
```

Este módulo imprime no *écran* um enquadramento de números, em torno do quadro, como guia para o utilizador.

MÓDULO 6.2.6

```
1370 REM ************************
1380 REM PRINT BOARD AND MOVE
1390 REM ************************
1400 PAPER 5: INK 1
```

```
1410 FOR I=1 TO 9: PRINT AT I*2-
1,1;"_ _ _ _ _ _ _ _ _ _ _ _
_ _"
1420 PRINT AT I*2,1;"_ _ _ _ _
_ _ _ _ _ _ _ _ _": NEXT I
1430 INK 1: PAPER 7
1440 PRINT AT 19,0;O$;O$;O$
1450 IF T=0 THEN RETURN
1460 PRINT AT M1,M2;CHR$ (143+D)
1470 IF A<>1 THEN RETURN
1480 IF T/E=INT (T/E) THEN PRINT
 AT 21,0;"RANDOM MOVE FOLLOWED"
1490 RETURN
```

Este módulo imprime no *écran* o quadro em que será efectuado o jogo. Imprime igualmente a seta relevante definida pelo utilizador para uma determinada jogada e informa o utilizador se estiver prestes a efectuar-se uma jogada casual.

Ensaio do módulo 6.2.6

Embora muitas das variáveis não tenham ainda sido definidas, se se instalar um STOP temporário na linha 1160, você deverá poder correr (RUN) o programa, inspeccionar as instruções, especificar um grau de dificuldade e ver o quadro, juntamente com a grelha que o envolve, impresso no *écran*.

MÓDULO 6.2.7

```
1160 REM *********************
1170 REM INPUT AND DIRECTIONS
1180 REM *********************
1190 INK 0: LET T=T+1: IF T>100
THEN CLS : PRINT AT 10,0;"SORRY-
CAN'T TAKE ANY MORE MOVES.": STO
P
```



```
1200 LET Q$=""
1210 PRINT AT 19,23;"MOVE ";T
1220 PRINT AT 21,17;"(0 FOR REVI
EW)"
1230 PRINT AT 19,1;"DOWN:";: INP
UT M1: PRINT M1
1240 IF M1>18 OR M1<0 THEN PRINT
 ">OUT OF RANGE": PRINT AT 19,0;
O$: GO TO 1210
1250: PRINT AT 20,0;O$: IF M1=0
THEN GO SUB 1660: GO TO 1210
1260 PRINT AT 21,0;O$;AT 20,1;"A
CROSS:";: INPUT M2: PRINT M2
1270 IF M2>30 OR M2<1 THEN PRINT
 ">OUT OF RANGE";AT 20,0;O$: GO
TO 1260
1280 PRINT AT 21,0;O$
1290 LET M(T,1)=M1: LET M(T,2)=M
2
1300 IF M1=P1 AND M2=P2 THEN GO
TO 1600
1310 LET M3=(M1<P1)-(M1>P1)+1
1320 LET M4=(M2<P2)-(M2>P2)+2
1330 LET D=M3*3+M4
1340 IF D>4 THEN LET D=D-1
1350 LET M(T,3)=D
1360 GO SUB 1370: GO SUB 1530: G
O TO 1190
```

Este módulo aceita o palpite do utilizador quanto à posição da presa e grava-o, para posterior revisão. É calculada a seta, definida pelo utilizador, apropriada para indicar a direcção da verdadeira posição.

Comentário

Linha 1290. Os movimentos do jogador são armazenados na disposição M.

Linhas 1310-1350. Estas linhas traduzem a direcção da presa, representada por P1 e P2, para a seta apropriada. Também esta é armazenada na disposição M, num local correspondente à jogada corrente.

Ensaio do módulo 6.2.7

Deverá agora poder definir um palpite quanto à posição da presa e deverá aparecer uma seta apontando, aproximadamente, para a direcção correcta... que pode ser verificada em referência a P1 e P2. O programa deverá agora aceitar jogadas que passem para além dos limites do quadro.

MÓDULO 6.2.8

```
1500 REM ************************
1510 REM MOVE INCREMENT
1520 REM ************************
1530 LET P1=P1+R1: LET P2=P2+R2
1540 IF T/E<>INT (T/E) THEN GO T
O 1570
1550 LET P1=INT (P1+RND*6+1): LE
T P2=INT (P1+RND*6+1)
1560 PRINT AT 21,0;"RANDOM MOVE"
1570 LET P1=P1+18*(P1<1)-18*(P1>
18)
1580 LET P2=P2+30*(P2<1)-30*(P2>
30)
1590 RETURN
```

Este módulo efectua o movimento da presa, incluindo um movimento casual, quando necessário.

Comentário

Linha 1530. P1 e P2 são as coordenadas da presa. R1 e R2 são os elementos da jogada casual, que são acrescentados a P1 e P2, em cada jogada, sendo colocados na secção das variáveis.

Linha 1540. A variável E é ajustada de acordo com o grau de dificuldade. Cada E altera-se sempre que é feita uma jogada casual, aumentando a dificuldade.

Linhas 1570-1580. Estas duas linhas asseguram que, se a presa sair do quadro, em qualquer direcção, reaparecerá do outro lado do quadro... Um efeito envolvente.

Ensaio do módulo 6.2.8

Deverá agora poder introduzir uma série de jogadas.

MÓDULO 6.2.9

```
1600 REM ************************
1610 REM SUCCESS AT LAST
1620 REM ************************
1630 PRINT AT 10,13;"GOT IT"
1640 INPUT "Another game? (Y/N)"
;Q$: IF Q$="Y" THEN GO TO 1030
1650 STOP
```

Este módulo anuncia o sucesso do jogador quando, ou se, a presa for encontrada.

Ensaio do módulo 6.2.9

Em vez de caçar a presa, corra (RUN) o programa e, quando lhe for pedido um palpite, pare (STOP) o programa e imprima P1 e P2 em modo directo. Faça correr novamente o programa com GOTO 1190 e introduza as coordenadas correctas, que deverão solicitar este módulo.

MÓDULO 6.2.10

```
1660 REM ************************
1670 REM REVIEW OF GAME
1680 REM ************************
1690 LET A=1: PRINT AT 19,0;O$;O
$;O$
1700 IF C1>=C THEN GO TO 1780
1710 PRINT AT 6,13;"REVIEW"
1720 PRINT AT 8,4;"REVIEW ALLOWA
NCE ";C;" MOVES"
1730 PRINT AT 10,8;"YOU HAVE USE
D ";C1
1740 PRINT AT 19,1;"LAST MOVE WA
S No.:";T-1
1750 PRINT AT 21,1;"INPUT FIRST
No. FOR REVIEW:";: INPUT T1: PRI
NT T1
1760 FOR J=T1 TO T-1
1770 LET C1=C1+1
1780 IF C1>C THEN PRINT AT 10,4;
"REVIEW RIGHTS EXHAUSTED": PAUSE
 200: GO TO 1860
1790 LET M1=M(J,1): LET M2=M(J,2
): LET D=M(J,3)
1800 GO SUB 1370
1810 PRINT AT 19,6;"REVIEW OF MO
VE:";J
```



```
1820 PRINT AT 21,20;"(0 TO QUIT)
"
1830 INPUT Q$: PRINT AT 19,0;O$;
O$;O$
1840 IF Q$="0" THEN GO TO 1860
1850 NEXT J
1860 LET T=T-1
1870 LET M1=M(T,1): LET M2=M(T,2
): LET D=M(T,3)
1880 GO SUB 1370
1890 LET A=0: RETURN
```

Poderá ter-se perguntado por que razão cada jogada, e a seta que lhe corresponde, são armazenadas na disposição M. Este módulo usa os dados armazenados em M para reconstituir jogadas anteriores e as pistas fornecidas para um reexame.

Ensaio do módulo 6.2.10

Deverá agora poder solicitar uma revisão de jogadas anteriores, quando lhe for pedida a coordenada inferior. Se este módulo estiver satisfatório, programa estará pronto a ser usado.

Sumário

Este jogo é um exemplo clássico de melhoramento. O jogo começou como uma coisinha insignificante, preparada para caber na memória de um *ZX81* de 1K. Foi divertido jogá-lo, pelo que o quadro foi alargado, para aumentar o desafio. Depois, por se ter tornado muito difícil, foi adicionada a função de revisão. Finalmente, a jogada casual completou o processo. A moral disto é que vale sempre a pena brincar com ideias simples, ainda que pareçam demasiado triviais, pois que, com alguma dissimulação, poderá ter nas mãos o sucessor do *Space Invaders*.

Avançando mais

1) Uma melhoria indiscutível seria um dispositivo de fim de jogo que permitisse a revisão das jogadas, juntamente com a posição da presa por cada jogada.

6.3 SELECÇÃO DE PALAVRAS

Se gosta de jogos de palavras e não se importa de esperar um par de horas, precisará de uma rotina de selecção de sequências. Trata-se de um eficiente método de selecção, capaz de reduzir o tempo necessário para colocar assuntos em ordem.

MÓDULO 6.3.1

```
1460 REM **********************
1470 REM SORT
1480 REM **********************
1500 LET A=INT (LN N/LN 2): LET
F=2^A-1
1510 LET F=INT (F/2): IF F=0 THE
N RETURN
1520 LET D=N-F: LET B=1
1530 LET A=B
1540 LET E=A+F: IF A$(A)<A$(E) T
HEN GO TO 1570
1550 LET B=B+1: IF B>D THEN GO T
O 1510
1560 GO TO 1530
1570 LET T$=A$(A): LET A$(A)=A$(
E): LET A$(E)=T$
1580 LET A=A-F: IF A<1 THEN GO T
O 1550
1590 GO TO 1540
```


Este módulo é alvo do mais minucioso estudo. Até agora utilizámos uma variedade de métodos para evitar material de selecção, preferindo encontrar o local correcto para um assunto de cada vez. Aquele local é armazenado numa disposição para indicadores, em vez da disposição principal que estiver a ser colocada na ordem correcta. Quando se tiver de introduzir um grande número de pequenos assuntos, isto pode revelar-se um processo pesado, com uma tediosa espera após cada acesso, enquanto não for encontrado o local correcto. Nestas circunstâncias é preferível introduzir os dados segundo a ordem por que forem aparecendo, seleccionando-os depois de todos os assuntos terem sido introduzidos.

Muitos programas usam uma selecção e muitas das selecções utilizadas em programas de fabrico caseiro são estupendamente ineficientes. Não se trata apenas de uma bela questão de estilo; uma selecção ineficiente num programa bastante simples pode, facilmente, revelar-se o processo mais moroso, necessitando de um cansativo período de espera, sempre que novos itens de dados são introduzidos. Um programa dedicado ao antecessor do *Spectrum*, o *ZX81*, publicado numa proeminente revista, levava vinte minutos a seleccionar cem itens de dados por ordem alfabética. O método usado neste módulo reduz esse tempo a três minutos.

A selecção empregue é conhecida como *Shell-Metzner*. Funciona esquadrinhando os dados a serem seleccionados e comparando pares que possam ser permutados se estiverem pela ordem errada. Para começar, a distância entre os assuntos de cada par é a maior potência de dois, a qual é menor que o número total de itens de dados. A selecção começa em 1 e compara o item de dados nessa posição com, digamos, o item na posição 64, no caso de haver 100 itens de dados. Os dados são esquadrinhados até que não haja mais pares possíveis separados por esta distância. O intervalo é então reduzido e começa a comparação de pares, desde o princípio.

Uma das melhores formas de compreender em pormenor o método de selecção é escrever os números 1 a 20 em pedaços de papel. Depois, com um bloco de apontamentos à mão para registar as variáveis, realizar a selecção, com base no módulo, tal como se fosse você o *Spectrum*. Quando tiver terminado poderá ainda achar difícil perceber por que razão este método funcionou. Mas o facto é que funciona e é uma das mais eficientes selecções disponíveis.

Comentário

Linhas 1500-1510. Estas duas linhas encontram a maior potência de dois, que é menor do que o número total de itens a serem seleccionados — de facto, é menor em um.

Linha 1520. D é um indicador; se a diferença corrente entre os itens de pares for F, então, o primeiro item do último par possível estará em N—F. B grava o primeiro item do par corrente.

Linha 1540. A$ está dimensionada para N itens. Note-se que os itens estão, realmente, a ser seleccionados no módulo pela ordem inversa, em extensão.

Linha 1570. Esta linha realiza a verdadeira permuta, se a linha 1540 encontrou os itens do par na ordem incorrecta.

Linha 1580. No caso de um item ter sido permutado para trás, por exemplo, do item 82 para o item 50, a sondagem retrocede para examinar o par de que ele é agora o segundo item, por exemplo, 18/50.

Ensaio do módulo 6.3.1

Não poderá ensaiar esta rotina até que tenha desenvolvido um programa que precise de utilizar uma selecção. O programa pode, facilmente, ser alterado para seleccionar dados numéricos, em vez de dados sequenciais, dimensionando uma variável A com N itens.

POSFÁCIO

Se penetrou em todos os programas deste livro, para além de merecer uma medalha é agora o orgulhoso possuidor de uma biblioteca de programas. É verdade que não se trata da maior biblioteca da história dos computadores, mas contém os instrumentos para desempenhar uma grande variedade de tarefas, para o que bastará adaptar os programas às suas necessidades específicas. Mais do que isso: é uma indicação, um indício — nem mais — do que o seu *Spectrum* aguarda proporcionar-lhe.

Os programas deste livro são ideias que me foram úteis, que me interessaram, que me resolveram problemas. Porque são minhas, algumas delas poderão falhar o alvo, quando chegar a altura de as aplicar aos seus problemas pessoais. Nesse caso, altere-as, mutile-as, parta-as e faça com elas peças sobressalentes. Ficarão muito melhores quando você as tiver tornado pessoais.

ÍNDICE

Os números após as entradas do índice referem-se aos números dos módulos relevantes.