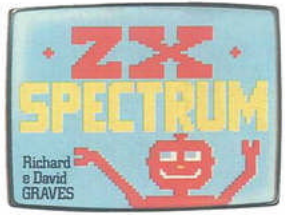
GUIA DO PRINCIPIANTE

Escrito num estilo simples e prático, este guia proporciona o primeiro passo da aprendizagem no uso do ZX Spectrum. Explica-se ao leitor como escrever programas simples, como usar variáveis numéricas e de texto, obter som, cores e formas gráficas.



Richard Graves, professor de informática, e David Graves, seu filho com onze anos de idade, acharam que muitos dos livros clássicos de introdução à programação eram demasiado difíceis. Juntos produziram um guia à programação elementar que pode ser usado por qualquer pessoa, de qualquer idade

Verbo

0-3

GUIA DO PRINCIPIANTE DO

GUIA DO PRINCIPIANTE

ZX
SPECTRUM

Richard
e David
GRAVES

sinclair
ZX Spectrum

Verbo

PROGRAMAÇÃO ELEMENTAR
FORMAS GRÁFICAS
SONS • JOGOS

Richard e David GRAVES

Verbo

Índice

1 Para começar 4
Como dar uma instrução simples; Como dar uma sequência de instruções; Três programas.

2 O computador como auxiliar 17
Matemática elementar; Variáveis numéricas; Outro tipo de variáveis numéricas; Variáveis alfanuméricas; Uso conjunto de variáveis numéricas e de texto.

3 Obtenção de som 35
Obtenção de efeitos especiais; Composição de melodias; Uso repetido de um grupo de linhas.

4 Palavras e planos de fundo coloridos 39
Uso de letras com cores diferentes na mesma linha; Letras intermitentes; Um modo diferente de colorir textos.

5 Desenhar com o computador 45
Desenhar semicircunferências; Desenhar toda a circunferência; Linhas coloridas; Reduzir o programa.

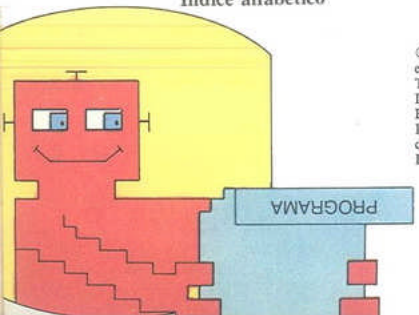
6 Desenhar também com blocos de cores 52
Pintar um único quadrado; Pintar uma fila de quadrados; Pintar várias filas de quadrados; Impressão de mais do que um bloco de cores.

7 «Fantasmas» 58
Nomes extensos para variáveis numéricas; Utilização de RND.

8 Gravação e leitura de programas 65

9 Glossário 68

Índice alfabético 72



Impresso em Março de 1985 por Empresa Litográfic do Sul, S.A.R.L. — Vila Real de Santo António
Dep. legal n.º 4674/84

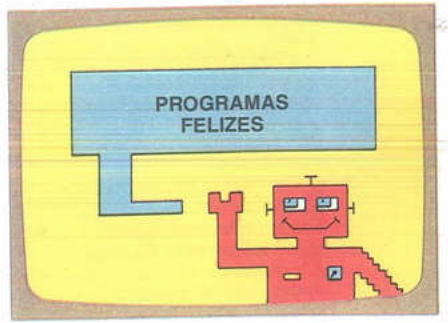
Introdução

Há tempo, a nossa família decidiu comprar um microcomputador. Por diversas razões, todos tínhamos interesse na aquisição. Um, por exemplo, esperava usá-lo para fins comerciais; outro queria jogar com ele. Mas, acima de tudo, acreditávamos que havia chegado o momento de a família entrar na idade do computador.

Ao princípio não nos apercebemos das dificuldades que nos esperavam. Mas logo se tornou claro que os livros clássicos de introdução ao *Basic* eram não só demasiado complicados para os mais jovens como também cheios de dificuldades para os mais idosos. Contudo perseverámos; e tendo aprendido à nossa custa, estamos desejosos de compartilhar este recém-adquirido conhecimento com outros principiantes.

Quando o leitor terminar este livro será capaz de escrever programas simples que usem variáveis numéricas e de texto, som, cores e formas gráficas simples. Esperamos que a sua leitura contribua para lhe abrir novos horizontes assim como escrevê-lo abriu os nossos. Também queremos agradecer a ajuda de Mary e Philip Graves.

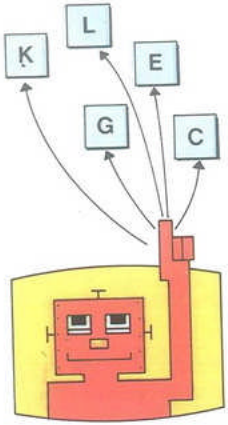
Richard e
David Graves



1 Para começar

Como dar uma instrução simples

O computador só pode executar o que lhe for ordenado. Dão-se-lhe as ordens escrevendo-as no teclado. Tudo o que se escreve no teclado aparece no *écran*. Cada palavra, letra ou símbolo aparece no ponto do *écran* onde há uma letra intermitente, chamada «cursor». O cursor pode ser qualquer das letras K, L, E, G ou C.



▲*Antes de se introduzir alguma coisa no computador* Spectrum, *tem de se ver um cursor (que acende e apaga) no* écran. *O cursor pode ser uma destas cinco letras. Cada cursor tem um uso diferente.*

? Problema 1: Quando se liga o computador não se vê qualquer cursor. Aparece unicamente a mensagem:

© 1982 Sinclair Research Ltd

na parte inferior do *écran*, e nada mais. Onde está o cursor?

✓ Premir a tecla ENTER, que está mesmo por cima da tecla BREAK SPACE na parte inferior direita do teclado. Agora sim, ver-se-á o cursor intermitente K. Tente-se depois introduzir uma instrução fictícia, por exemplo "Toca", e escrever o seguinte no teclado:

Toca «Olha para tras»

Mas... mal se carrega na tecla T, acontece o inesperado. Em vez de surgir um T no *écran*, aparece:

RANDOMIZE

seguido por um cursor intermitente L.

? Problema 2: É fácil cometer erros ao introduzir uma instrução.

✓ Repare-se na parte inferior esquerda do teclado, onde está a tecla marcada CAPS SHIFT. A tecla zero está na parte superior direita do teclado. Esta tecla está marcada com um Ø. Não confundir com a tecla O, que é a maiúscula da letra 'o' do alfabeto. De aqui em diante, todo o zero será escrito Ø para facilitar a distinção. Por cima da tecla Ø ver-se-á a palavra DELETE, que significa 'apagar' ou 'desfazer-se de'. Mantenha um dedo na tecla CAPS SHIFT. Se se continuar a carregar nela enquanto se prime Ø, desaparece, de cada vez que isso se faz, a palavra, letra ou símbolo que estiver à esquerda do cursor. Assim, se se cometer um erro e se quiser emendá-lo, prime-se CAPS SHIFT e Ø até apagar o erro e depois escreve-se a instrução correcta. Pode-se agora apagar a palavra RANDOMIZE, mas como fazer para que o computador toque a melodia intitulada «Olha para trás»?

? Problema 3: O computador compreende unicamente instruções escritas numa linguagem especial, chamada *Basic*.

✓ Sim! Felizmente, o computador *Spectrum* oferece ajuda suficiente para tornar esta linguagem relativamente fácil de aprender. Por exemplo, o computador sabe que no início de uma instrução deve estar um comando em *Basic*. Deste modo, no início de cada nova instrução, o computador mostra um cursor intermitente K. (Diz-se que está no modo K.) Comandos são, simplesmente, palavras em *Basic* que se usam muitas vezes.

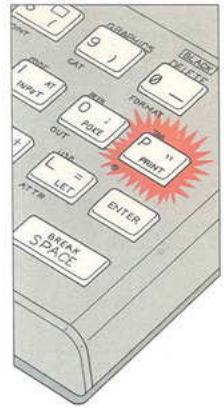
Se se premir uma tecla alfabética quando o computador estiver no modo K, não se verá essa letra no *écran* mas, sim, o comando que se mostra 'a branco' no canto inferior direito da tecla. Por isso quando se premiu T, o computador imprimiu o comando RANDOMIZE. (Só está escrito RAND nessa tecla, porque não há espaço suficiente para escrever o comando completo.) Se se observarem todas as teclas alfabéticas, não se encontrará nenhuma com o comando TOCA. TOCA não é uma instrução em *Basic*. Mas na tecla P existe o comando PRINT, que se poderá utilizar em vez do anterior. Sempre que se queira que o computador imprima uma mensagem no *écran*, deve premir-se o comando PRINT seguido de " (aspas). Agora escreve-se a mensagem. E finaliza-se com outras aspas. Resumindo, é necessário escrever:

PRINT "Olha para tras"

Observe o cursor K e recorde que, ao começar, está no modo K. Agora carregue na tecla P para obter o comando PRINT. Se olhar para o *écran*, verá

PRINT

seguido de (surpresa!) um novo cursor intermitente. Já

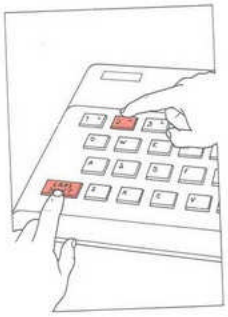


▲*A instrução PRINT é necessária para a impressão das mensagens no* écran.

não se encontra no modo K . E o computador dá uma nova ajuda! Ele sabe que, uma vez que se começou uma instrução com um comando, não se quererá usar outro comando a seguir. O novo cursor intermitente é L . Está agora no modo L . Quando se está no modo L , podem escrever-se letras. Se quisermos grafar letras *minúsculas* bastará premirmos teclas alfabéticas. Mas se quisermos escrever uma *maiúscula*, deveremos manter um dedo na tecla CAPS SHIFT e premir a letra desejada com outro dedo. Experimente, leitor!

(Mais adiante, nesta obra, o leitor talvez necessite escrever uma mensagem em letras maiúsculas. A fim de evitar manter um dedo sempre na tecla CAPS SHIFT, prima com outro dedo a tecla com o número 2, na parte de cima do teclado. Por cima desta tecla diz CAPS LOCK em letras brancas. Ensaie e verifique que o cursor agora é um C intermitente. No modo C , qualquer tecla alfabética que seja premida será maiúscula, não sendo necessário manter premida a tecla CAPS SHIFT para escrever uma mensagem com letra grande! Para regressar ao modo L basta voltar a premir as teclas CAPS SHIFT e 2 simultaneamente. Apague, com DELETE, todas as letras que haja escrito enquanto praticava com o modo C , de tal modo que fique unicamente no *écran* o comando PRINT.)

Às vezes, é necessário dar espaços entre as letras ou entre as palavras da mensagem (recorde-se que uma mensagem é tudo o que está entre aspas). Para isso basta premir a tecla BREAK SPACE, no canto inferior direito do teclado. Fora das aspas, os espaços entre quaisquer palavras ou símbolos são adicionados automaticamente pelo próprio computador.



▲ *Para teclar toda a mensagem em letras maiúsculas, mantém-se um dedo na tecla CAPS SHIFT enquanto, com outro dedo, se prime uma só vez a tecla numérica 2.*

? Problema 4: Depois de PRINT, deve escrever-se o símbolo ", que está no canto superior direito da tecla P, mas... *a vermelho*.

✓ Quando se está em qualquer dos modos K , L ou C e se necessita escrever um dos símbolos ou palavras marcadas a vermelho, tanto nas teclas numéricas como nas alfabéticas, dever-se-á manter um dedo na tecla SYMBOL SHIFT (ao lado esquerdo da tecla BREAK SPACE) e premir a tecla desejada com outro dedo. Pronto, o leitor já sabe tudo o que é preciso para escrever:

PRINT "Olha para tras"

Para emendar um erro cometido, o leitor deve usar a tecla DELETE e apagar o que está errado, e escrever de novo o que falta à direita do cursor. Depois de teclar uma instrução qualquer não acontece nada enquanto não se carrega na tecla ENTER, próxima do canto inferior direito do teclado. Por isso, prima ENTER. Agora pode ler a sua mensagem na parte superior do *écran:*

Olha para tras

Ao mesmo tempo aparece outra mensagem no canto inferior esquerdo do *écran:*

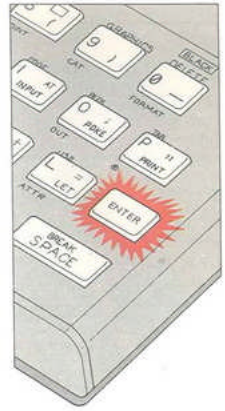
0 OK, 0:1

significando que está tudo bem! Note-se que o cursor desapareceu do *écran*. Para fazê-lo reaparecer basta premir ENTER ou qualquer outro comando. Se se tivesse cometido algum erro e se premisse ENTER, a instrução não teria sentido em *Basic* e portanto a mensagem não apareceria na parte superior do *écran*. Em lugar disso, a instrução teria ficado no mesmo sítio e um ponto de interrogação intermitente assinalaria, geralmente no local do erro, que qualquer coisa não andava bem. Corrigir-se-ia então o erro como habitualmente (usando DELETE) e, finalmente, premir-se-ia ENTER outra vez. Para ver como é, o leitor escreva no teclado o seguinte:

PRINT "Hoje e quinta-feira

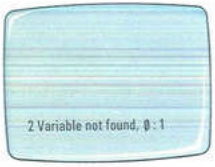
e prima ENTER. Deveria ter encerrado a mensagem com aspas e, como não o fez, a instrução não foi executada, ficando onde estava mas aparecendo um ponto de interrogação intermitente depois da palavra quinta-feira. O erro é no fim da linha e, por isso, não é preciso apagar (DELETE) nada. Basta adicionar as aspas que faltam e desaparecerá o ponto de interrogação intermitente. Agora tudo está certo nesta linha, de modo que se pode premir ENTER para executá-la.

Se o erro for de palmatória, o computador poderá decidir que será melhor repetir tudo desde o princípio. Ensaie esta instrução:



▲ *Depois de teclar uma instrução, deve carregar-se na tecla ENTER. Só assim o computador fará o que se lhe pediu, e imprimirá uma mensagem dizendo que recebeu essa instrução.*

▼*Quando se tenta introduzir uma instrução com um erro grave, o computador apaga toda essa linha e, em seu lugar, imprime uma mensagem informando que alguma coisa não está bem.*



PRINT Estou bem, obrigado

Agora prima ENTER. Desta vez, esqueceu-se do par de aspas e o computador apagou toda a linha e imprimiu a seguinte mensagem no fundo do *écran:*

2 Variable not found, 0:1

Agora, para repetir a instrução anterior, é necessário premir ENTER para limpar o *écran*. Agora, o leitor já sabe introduzir uma instrução simples no computador. Antes de prosseguir, ensaie algumas instruções PRINT que tenha imaginado. Por exemplo, peça ao computador que imprima no *écran* o nome do país que mais gostaria de visitar.

Como dar uma sequência de instruções

Já se aborreceu executando uma só instrução? Se quisermos que o computador faça alguma coisa mais interessante, será necessário dar-lhe uma lista de instruções.

? Problema 5: O computador tenta executar qualquer instrução desde que se prima ENTER. Como fazê-lo esperar o tempo suficiente para introduzir uma lista de instruções?

✓ Escrevendo um número antes de cada instrução, ou linha! O computador guarda então cada linha numerada na sua memória. Comece com o número 1Ø para a primeira linha, 2Ø para a segunda, etc. Quando o computador estiver nos modos K ou L, poderão escrever-se números premindo simplesmente as teclas numéricas no topo do teclado.

Se se começar uma instrução com um número, o computador continuará no modo K. E como sabe que um número no início duma instrução é um número de linha, o computador ainda ficará à espera de um comando. Escreva-se agora uma sequência de instruções, sem esquecer premir ENTER depois de cada linha. Se a instrução estiver correcta, aparecerá na parte superior do *écran*, onde lentamente se construirá a sequência de instruções correctas.

Por vezes as instruções não cabem numa só linha do *écran*. Não é preciso carregar na tecla ENTER quando já não houver espaço para escrever nessa linha. Continua-se a teclar: algumas palavras passarão para a linha seguinte do *écran*, mas o computador compreenderá que tudo faz parte da mesma linha, porque ainda não se premiu a tecla ENTER.

Os símbolos . (ponto final) e ? (ponto de interrogação) estão a vermelho, nas teclas alfabéticas M e C; o - (hífen) e o ! (ponto de exclamação) estão nas teclas alfabética J e na numérica 1. É preciso usar a tecla SYMBOL SHIFT para obter qualquer deles.

```
1Ø PRINT "Sou o computador Spectrum ZX."
2Ø PRINT "Espero que se divirta comigo."
3Ø PRINT "Sabe como se formou a palavra"
4Ø PRINT "BASIC?"
5Ø PRINT "Do ingles: Begginers All-purpose"
6Ø PRINT "Symbolic Instruction Code!"
7Ø STOP
```

A frase em inglês cujas primeiras letras formam a palavra *Basic* significa «Código de instruções simbólicas para principiantes e para todos os fins».

O comando STOP na linha 6Ø diz ao computador que terminou a lista de instruções. Encontra-se a vermelho na tecla A, por isso o leitor deve carregar com um dedo na tecla SYMBOL SHIFT e, mantendo-a premida, carregar com outro dedo na tecla A.

? Problema 6: Agora que se teclou toda a lista de instruções, como se obriga o computador a executá-las?

✓ Dando o comando RUN (premindo a tecla R) e carregando em ENTER. Uma lista de instruções é um PROGRAMA. Se o leitor seguiu estas instruções, acaba de introduzir e executar o seu primeiro programa. Próximo da parte inferior do *écran* verá uma mensagem dizendo-lhe que o computador terminou a execução do programa na linha 6Ø:

9 STOP statement, 6Ø:1

Se deseja consultar outra vez a lista de instruções, deverá dar o comando LIST (premindo a tecla K) e carregar em ENTER.

Para correr o programa outra vez, prime-se RUN novamente, seguido de ENTER.

▲*Depois de introduzir uma lista de instruções, prime-se RUN para que o computador as cumpra.*

? Problema 7: Como libertar-se de um programa sem desligar o computador?

✓ Prime-se o comando NEW (na tecla A) e carrega-se em ENTER. Deve dar-se este comando antes de introduzir outro programa. Agora um desafio: escreva o leitor um programa, usando a instrução PRINT. Por exemplo, imagine um programa para imprimir no *écran* os títulos dos seus livros favoritos. Ou um programa que imprima uma lista de cinco animais selvagens.

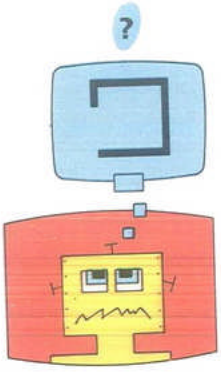
Três programas que mostram o que o computador pode fazer

Escreva os programas seguintes e repare no que acontece. Demonstrar-lhe-ão as possibilidades do computador. (Depois de cada programa, dão-se sugestões valiosas para encontrar certas teclas.)

PROGRAMA UM

```
10  PLOT   10,48
20  DRAW   100,0
30  DRAW   0,100
40  DRAW   -100,0
50  DRAW   0,-100
```

O comando PLOT está na tecla Q e o comando DRAW na tecla W. A , (vírgula) está a vermelho na tecla N. O - (sinal menos) é dado pelo hífen na tecla J. Agora, execute (RUN) o programa. Se teclou correctamente todas as linhas, verá um quadrado no *écran*.



▲ *Se alguns números do programa 'Um' forem mal escritos, o quadrado obtido no écran não será perfeito.*

? Problema 8: Se, depois de executar o programa, o quadrado não estiver completo, ou aparecer distorcido, é porque, provavelmente, foram teclados incorrectamente alguns números.

✓ Prime-se LIST e verificam-se cuidadosamente todas as linhas. Quando se encontrar o erro, tecla-se de novo toda a linha e prime-se ENTER. O computador substituirá automaticamente a linha incorrecta pela nova linha acabada de introduzir. Executando o programa novamente e se não houver mais erros, aparecerá um quadrado perfeito.

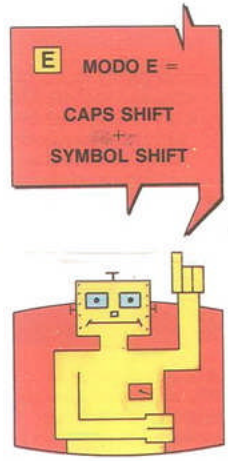
Agora prima-se LIST outra vez e adicionem-se as linhas seguintes para pintar o quadrado de magenta (púrpura-claro):

```
60  INK   3
70  FOR   x=3 TO 15
80  FOR   y=13 TO 25
90  PRINT  AT x,y;"■"
100 NEXT  y
110 NEXT  x
120 STOP
```

O comando FOR está na tecla F e NEXT na tecla N. O comando AT está a vermelho na tecla I. O comando TO encontra-se a vermelho na tecla F. O = (sinal de igual) está a vermelho na tecla L. O ; (ponto e vírgula) está a vermelho na tecla O. Até aqui foi tudo fácil. Mas que fazer com INK na linha 60 e com o quadrado preto ■ na linha 90?

? Problema 9: O comando INK na linha 60 não está em nenhuma tecla. Encontra-se por baixo da tecla X, a vermelho. Como obtê-lo?

✓ Existem muitas palavras e símbolos *extras* por cima (a verde) e por baixo (a vermelho) das teclas alfabéticas. Para obtê-las, é preciso primeiro passar o cursor ao modo [E] . Para isso, deve-se premir com um dedo a tecla CAPS SHIFT, enquanto se prime com outro dedo, e uma vez só, a tecla SYMBOL SHIFT: o cursor [K] ou [L] mudará para um [E] intermitente. Passou-se ao modo [E] , também chamado muitas vezes *Modo alargado*. Agora, estando no modo [E] , se se carregar numa tecla alfabética escrever-se-á realmente a palavra ou símbolo que estiver *a verde* por cima dela. E premindo uma tecla alfabética enquanto com outro dedo carregar na tecla CAPS SHIFT obtém-se a palavra ou símbolo que estiver *a vermelho* por baixo dessa tecla alfabética. Assim, para teclar INK, tem de se passar ao modo [E] , e só então premir a tecla X enquanto outro dedo mantém carregada a tecla CAPS SHIFT. O modo [E] dá unicamente para uma, e só uma, pressão de uma tecla alfabética, passando automaticamente ao modo [L] depois de premida essa tecla. Se for necessário voltar ao modo [E] , terá de se repetir todo o procedimento.



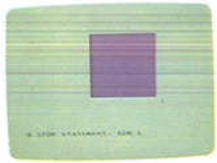
? Problema 10: O símbolo quadrado ■ que aparece a preto na linha 90 é uma forma ou símbolo *gráfico*.

Está na tecla numérica 8. Como se obtém (e também os outros símbolos gráficos das teclas 1 a 7)?

☑ Para teclar um símbolo gráfico é necessário primeiro passar ao modo G. Faz-se isto mantendo um dedo a premir a tecla CAPS SHIFT, enquanto outro dedo carrega uma, e uma vez só, na tecla 9. Para que se lembre que a tecla 9 é a necessária para passar ao modo G, a palavra GRAPHICS está escrita a branco por cima da tecla 9. Logo que se passa ao modo G, o cursor intermitente muda de K ou L para G.

Agora, antes de teclar um dos símbolos gráficos, é necessário decidir se se quer escrevê-lo exactamente como aparece na tecla, com as áreas brancas a branco e com as pretas a preto, ou se se prefere o *oposto*, isto é, com as áreas brancas a preto e as áreas pretas a branco. Para obtê-los como aparecem na tecla basta premir a tecla numérica desejada. Para conseguir o oposto, mantém-se a tecla CAPS SHIFT premida com um dedo enquanto outro carrega na tecla numérica. Assim, para obter o que se quer é preciso passar ao modo G, e depois manter um dedo carregando na tecla CAPS SHIFT enquanto outro prime a tecla 8. Continua-se no modo G até dizer ao computador para sair dele. Consegue-se sair do modo G mantendo um dedo a carregar na tecla CAPS SHIFT enquanto outro dedo prime novamente a tecla 9. Então passa-se ao modo K ou L.

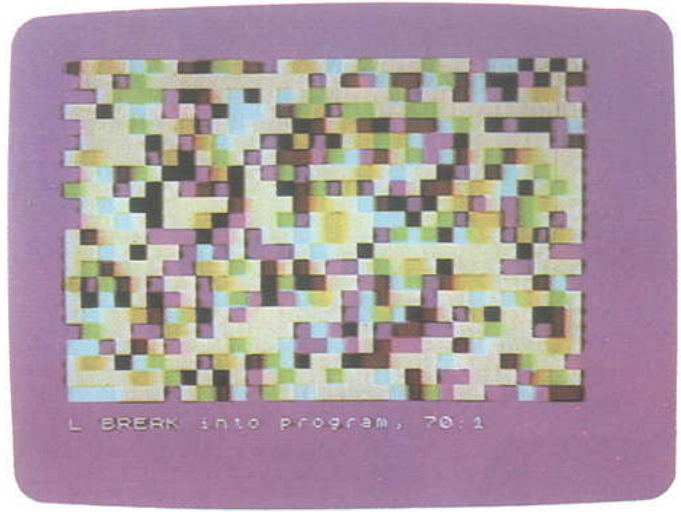
Executando agora o programa, ver-se-á que o quadrado fica cheio de cor. Para limpar este programa, prime-se NEW seguido de ENTER. (Neste momento, o leitor deve estar a dizer para si mesmo que o teclado do *Spectrum* é bastante difícil para um principiante. Não se preocupe, com a prática torna-se muito mais fácil.)



▲*Usa-se a instrução INK, juntamente com os símbolos gráficos nas teclas 1 a 8, para colorir zonas do* écran.

PROGRAMA DOIS

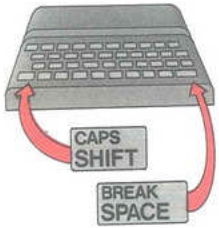
```
10  LET   x=INT (RND*22)
20  LET   y=INT (RND*32)
30  LET   z=INT (RND*7)
40  BORDER  1
50  INK   z
60  PRINT   AT x,y;"■"
70  PAUSE   5
80  GOTO   10
90  STOP
```

O comando LET encontra-se na tecla L, e o comando BORDER na tecla B. O comando PAUSE, na tecla M e o GOTO na tecla G. INT está a verde *acima* da tecla R, e RND também *acima* da T. Por isso, é preciso passar ao modo E. O símbolo * está a vermelho na tecla B, e o (e o) (parênteses) estão a vermelho nas teclas 8 e 9. (Recordemos: Tem de se passar ao modo E e de manter o dedo carregando na tecla CAPS SHIFT para obter INK.)

Este programa imprime uma série sem fim de quadrados coloridos em fundo branco, rodeado por uma raia azul-escura. Logo na primeira vez que o executamos, notar-se-á que não aparecem quadrados numa zona estreita, ao fundo da área branca. Mais adiante damos a explicação deste facto. O programa segue uma e outra vez, sem nunca parar, porque ao chegar à linha 80 o computador executa a instrução GOTO 10. Esta diz-lhe para seguir imediatamente para a linha 10 e, por isso, executa o programa novamente numa situação de círculo vicioso. A linha

▲*Este programa nunca termina, por causa da instrução GOTO 10, e continuará a imprimir quadrados de cores diferentes no* écran.



▲ *Para terminar um programa que está a ser executado, prime-se estas duas teclas ao mesmo tempo.*

90 nunca será executada. É uma situação conhecida como um ciclo *(loop)* sem fim.

? Problema 11: Como parar a execução dum progama, sem desligar o computador?

☑ Mantendo um dedo carregando na tecla CAPS SHIFT, prime-se com outro dedo a tecla BREAK SPACE. A execução do programa pára e aparece na fresta ao fundo do *écran* uma mensagem semelhante a esta:

```
L BREAK into program, 20:1
```

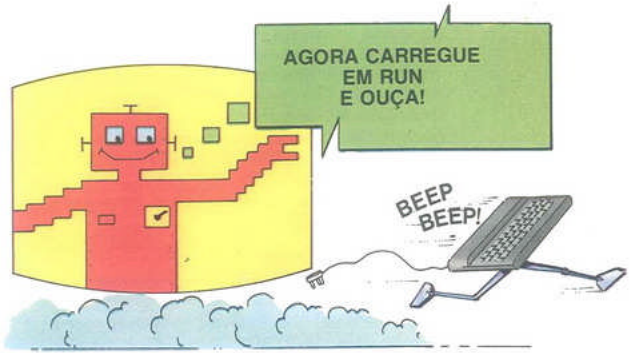
Nesta mensagem, o número 20 é o número da linha onde parou a execução do programa. Poderia ter sido um dos outros números de linha. Se agora se executar o programa outra vez, obtém-se um padrão diferente do anterior. Ou, se for desejável o mesmo padrão, dá-se o comando CONTINUE (CONT na letra C) seguido de ENTER. O computador continuará então a execução exactamente a partir do ponto onde fora interrompida.

Note-se que imediatamente depois da interrupção do programa, a fresta branca próxima do fundo do *écran* desapareceu. Por isso, quer se execute o programa novamente ou quer se opte pela continuação, a imagem fica com melhor aspecto. Quando nos quisermos desembaraçar deste programa, primem-se as teclas CAPS SHIFT e BREAK SPACE, depois escreve-se NEW seguido de ENTER.

PROGRAMA TRÊS

```
10  BEEP  1,0
20  BEEP  .5,2
30  BEEP  .5,0
40  BEEP  1,5
50  BEEP  1,4
60  BEEP  .5,2
70  BEEP  .5,4
80  BEEP  1,5
90  GOTO  10
100 STOP
```

O comando BEEP encontra-se por debaixo da tecla Z. Por isso é preciso passar ao modo [E] e depois manter um dedo na tecla CAPS SHIFT enquanto outro dedo prime a tecla Z. (Recordemos que o ponto final está na tecla M.) Como se notará depois de executar este programa, o computador também pode emitir sons!



Revisão sumária

Até agora temos andado a descobrir os segredos do teclado. Agora o leitor já sabe:

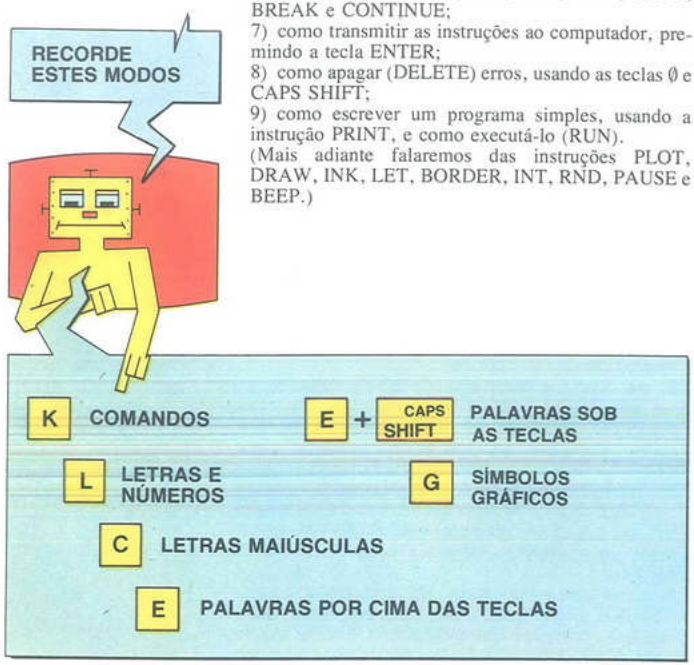
1) que começa no modo [K], se premir teclas numéricas escreverá números; se premir as alfabéticas, escreve comandos a branco;

2) que depois se passa ao modo [L], podendo escrever números ou letras conforme carregar em teclas numéricas ou alfabéticas (e que se quiser escrever muitas *maiúsculas* seguidas poderá passar ao modo [C]);

3) como obter os símbolos ou palavras a vermelho marcados *na própria tecla* quando se está nos modos [K] ou [L]. Basta premi-las com um dedo enquanto outro dedo mantém carregada a tecla SYMBOL SHIFT;

4) como passar ao modo [E], premindo a tecla SYMBOL SHIFT uma única vez, enquanto mantém um dedo carregando na tecla CAPS SHIFT. Agora pode obter as palavras a verde *acima* das teclas

alfabéticas simplesmente premindo essa tecla e as palavras a vermelho *abaixo* delas obtêm-se premindo essa tecla mas mantendo outro dedo carregando na tecla CAPS SHIFT ou SYMBOL SHIFT;
5) como entrar e sair do modo [G], premindo a tecla 9 enquanto outro dedo mantém carregada a tecla CAPS SHIFT. Uma vez no modo [G], pode obter os símbolos gráficos marcados nas teclas numéricas simplesmente premindo essas teclas e também obter os opostos desses símbolos premindo as teclas numéricas enquanto outro dedo mantém carregada a tecla CAPS SHIFT.
E aprendeu ainda mais:
6) como usar nove instruções muito úteis:
PRINT, SPACE, STOP, RUN, LIST, NEW, GOTO, BREAK e CONTINUE;
7) como transmitir as instruções ao computador, premindo a tecla ENTER;
8) como apagar (DELETE) erros, usando as teclas Ø e CAPS SHIFT;
9) como escrever um programa simples, usando a instrução PRINT, e como executá-lo (RUN).
(Mais adiante falaremos das instruções PLOT, DRAW, INK, LET, BORDER, INT, RND, PAUSE e BEEP.)



2 O computador como auxiliar

Matemática elementar

Primeiro que tudo, tente-se encontrar os sinais + (mais) e - (menos). O + está a vermelho na tecla K, por isso é preciso manter um dedo na tecla SYMBOL SHIFT enquanto se prime a tecla K, para obtê-lo. O - está também a vermelho, mas na tecla J.

Eis uma soma elementar: 3+4. Se se teclar 3+4 e premir ENTER, nada sucederá excepto o aparecimento de um ponto de interrogação intermitente depois do sinal +. Terá que se apagar (DELETE) esta linha. Mas se se der a instrução apropriada, o computador fará a soma muito rapidamente e imprimirá o resultado na parte superior do *écran*, dando também a sua mensagem usual 'OK' no fundo. Para ensaiar, escreva-se:

```
PRINT  3+4
```

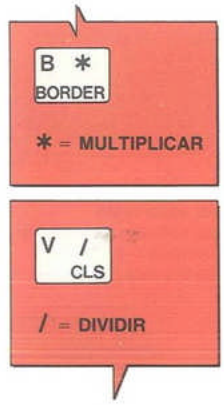
premindo ENTER. A resposta foi dada imediatamente! Grande coisa, pensará o leitor, que também o fez, e até antes dele. Eis uma conta mais difícil: 4×8989898.

? Problema 12: Foi fácil encontrar os sinais mais e menos. Mas onde estão os de × (multiplicação) e de ÷ (divisão)?

☑ O sinal de multiplicação em *Basic* é *, e está a vermelho na tecla B. O sinal de divisão é / (está a vermelho na tecla V). Deve teclar-se agora:

```
PRINT  4*8989898
```

premindo ENTER. A resposta aparece num abrir e fechar de olhos: 35959592. Experimente obter os resultados das contas seguintes:



▲*Para introduzir estes símbolos, mantém-se um dedo na tecla SYMBOL SHIFT, enquanto se carrega, com outro dedo, nas teclas B ou V.*

5×12345678
1471365÷9

▲ *Introduzindo uma soma com mais do que uma parcela, o computador calculará primeiro as multiplicações e as divisões.*

▲ *Mas se parte da soma foi posta entre parênteses, ele calculará primeiro o que estiver dentro dos parênteses.*

Se dermos uma expressão com vários termos, tal como 2×5-4÷2, o computador fará primeiro as multiplicações e divisões. Assim calculará 2×5=10, 4÷2=2 seguido de 10-2=8. O resultado é 8. Se a expressão tiver parênteses (os símbolos vermelhos nas teclas 8 e 9), o computador calculará primeiro o que estiver DENTRO DOS PARÊNTESES. Assim 2×(5-4)÷2 será calculado 5-4=1, depois 2×1=2 e finalmente 2÷2=1, portanto o resultado final é 1. Verificam-se estes resultados introduzindo as seguintes instruções, e premindo ENTER no fim de cada uma:

```
PRINT  2*5-4/2
PRINT  2*(5-4)/2
```

Usa-se o ponto final (na tecla M) como ponto decimal, em qualquer expressão que contenha números decimais. Ensaie o leitor:

```
PRINT  35*7.25
```

e prima ENTER. O resultado deverá ser 253.75. Agora experimente calcular algumas expressões inventadas por si. Por exemplo, divida o número de alunos da sua escola pelo número de professores para calcular quantos alunos há por professor.

Variáveis numéricas

(Usam-se cada vez que se calcula o mesmo tipo de contas, mas com números diferentes.)
Imagine o leitor que o seu aniversário está próximo e que pediu aos seus amigos que lhe oferecessem dinheiro, em vez de presentes. Suponha que decidiu gastar um quarto do dinheiro que lhe derem comprando selos para a sua colecção, pôr outro quarto no mealheiro e com a metade que resta, comprar uns discos *pop*. Estas quantidades são *constantes*. Mas o leitor não tem ideia quanto dinheiro lhe oferecerão! Pode ser qualquer quantia desde 1 escudo até 5000 escudos, ou talvez mais. Como a quantidade não é constante, chama-se-lhe uma *variável*. No programa que segue, representa-se esta *variável numérica* pela letra 'a'. (E manter-se-á *fora* das aspas nas instruções PRINT.)

```
10  PRINT  "Deram-me Esc."; a
20  PRINT  "Para selos, tenho Esc."; a/4
30  PRINT  "Para guardar, tenho Esc."; a/4
40  PRINT  "Para discos, tenho Esc."; a/2
50  STOP
```

Note que é preciso haver um ponto e vírgula entre as aspas duma mensagem e uma variável numérica. Também se poderia ter usado uma vírgula em vez dum ponto e vírgula, mas se assim fosse, ao imprimir no *écran*, teria ficado um espaço muito grande entre Esc. e a quantia em dinheiro. Um ponto e vírgula diz ao computador que o próximo valor a sair tem que ser impresso imediatamente após o último carácter lançado pelo computador no *écran*. Agora execute (RUN e ENTER) este programa. No *écran*, só verá impresso:

```
Deram-me Esc.
```

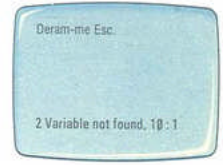
e cerca da parte inferior do *écran*, a mensagem:

```
2 Variable not found, 10:1
```

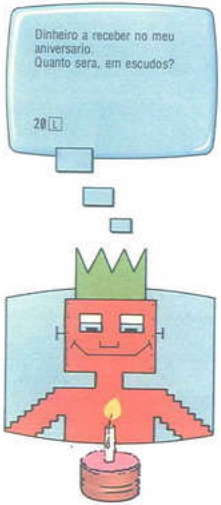
O que significa que não foi encontrada uma *variável*. Foi introduzida uma variável 'a', mas o leitor não teve oportunidade de dizer ao computador qual o valor que 'a' teria! Ou, dito de outra forma, a variável 'a' não foi definida. Suponha que recebe esc. 2000 no seu aniversário. Adicione ao programa a linha:

```
7  LET  a=2000
```

Encontra o comando LET na tecla L. (Agora se vê porque se numeram as linhas de um programa de 10 em 10. Assim há sempre possibilidade de introduzir novas linhas entre outras já existentes.) Se executar (RUN e ENTER) o programa neste momento, o computador imprimirá:



▲ *Para que o computador diga quanto dinheiro lhe deram, o leitor deve definir primeiro quanto vale a variável da linha 10 (a letra A).*



▲*Usando uma instrução INPUT, o programa faz o computador esperar até que seja introduzido o dinheiro que o leitor pensa que lhe irão dar. Depois ele calculará quanto poderá gastar em cada artigo.*

```
Deram-me Esc. 2000
Para selos, tenho Esc. 500
Para guardar, tenho Esc. 500
Para discos, tenho Esc. 1000
```

e, em baixo, verá a mensagem:

```
9 STOP  statement, 50:1
```

Mas com este método, deverá escrever no teclado uma linha 7 cada vez que ensaie uma quantia diferente! Isto não é muito prático, por isso é melhor apagar a linha 7, e tentar outro método.

? Problema 13: Como apagar uma linha já introduzida no computador, ou, dito de outra forma, como apagar uma instrução depois de premir ENTER?

☑ Para apagar a linha 7, e não querendo substituí-la por outra, deve-se escrever o número 7 e carregar em ENTER. Querendo apagar outra linha, bastaria escrever o número dessa linha e premir ENTER. Se fosse uma substituição, deveria escrever-se a nova linha com o mesmo número daquela que se queria substituir. Ao premir ENTER, o computador imediatamente apagaria a linha antiga e poria, em seu lugar, a nova. Apagando agora a linha 7, observa-se que ela desapareceu do *écran*. Introduza as novas linhas seguintes:

```
4 PRINT  "Dinheiro a receber no meu ani-»
5 PRINT  "versario."
6 PRINT  "Quanto sera, em escudos?"
7 INPUT  a
```

O comando INPUT está na tecla I. Execute o programa. No *écran* aparece:

```
Dinheiro a receber no meu ani-
versario.
Quanto sera, em escudos?
```

Aqui o computador fica à espera, sem fazer nada, por causa da instrução INPUT a. E espere até que se tecle o número a dar à variável 'a' e só continua a execução quando se premir ENTER. A esta operação chama-se uma *entrada de dados*, e diz-se também que se introduziu um dado para ser guardado na memória chamada 'a'. Agora tente o leitor introduzir o dado 2000 (basta digitar 2000, que aparece ao fundo do *écran*, e carregar em ENTER). O computador continuará imediatamente a execução do programa e imprimirá os resultados anteriores. E agora execute o programa outra vez (RUN e ENTER). Introduza o dado 3200. Os resultados serão, naturalmente, diferentes. Na realidade, pode executar o programa quantas vezes quiser, introduzindo uma quantia de dinheiro sempre diferente e vendo, para cada hipótese, quanto pode amealhar e quanto pode gastar.

? Problema 14: Como dar um título ao programa?

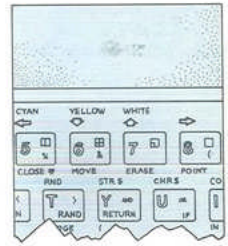
☑ Para dar um título a um programa qualquer, deve-se introduzir uma linha com o comando REM (na tecla E) seguido do título entre aspas. Esta linha deve aparecer antes de todas as outras, portanto dá-se-lhe um número que seja menor do que qualquer das outras linhas. Digamos que este programa se intitula *Parabéns;* neste caso, o leitor deverá introduzir:

```
3 REM  "PARABENS"
```

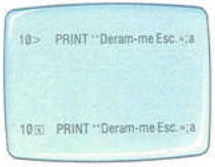
O computador não actua quando encontra a palavra REM. Por isso o comando REM é útil para introduzir linhas que expliquem a aplicação e uso desse programa. O leitor apreciará estas linhas de comentários quando examinar um programa escrito dias ou meses antes. O programa apresentado está agora completo e funcionando bem. Mas, antes de dar o comando NEW e seguir para um assunto diferente, use o leitor o programa para resolver o problema seguinte.

? Problema 15: Deve haver uma forma de alterar só uma pequena parte duma linha já introduzida, para que não seja necessário repetir toda a linha. Mas como?

☑ Se, por exemplo, se quer substituir o ponto e vírgula da linha 10 por uma vírgula, escreve-se LIST e prime-se ENTER de maneira a ter o programa "PARABENS" diante de nós, no *écran*. Agora observe-se a primeira fila do teclado do *Spectrum*. Por cima das teclas 5, 6, 7 e 8, vêem-se setas apontando para a esquerda, para baixo, para cima e para a direita. Com estas setas o cursor move-se para qualquer posição do *écran*, para se alterar linhas do programa. Chama-se *editar* a estas alterações. Para usar as setas basta manter um dedo carregando a tecla CAPS SHIFT e com outro dedo, premir a tecla por *cima* da qual está a seta desejada.



▲*Usando as teclas numéricas 5 a 8 move-se o cursor do programa no écran, nos sentidos que as setas mostram.*



▲ *Para fazer alterações numa linha já introduzida é preciso primeiro fazer uma cópia dessa linha no fundo do écran. Para fazer a cópia, posiciona-se o cursor do programa entre o número da linha e a primeira instrução da linha pretendida. Depois mantém-se um dedo carregando na tecla CAPS SHIFT e, com outro, prime-se a tecla 1.*

Mantendo um dedo na tecla CAPS SHIFT e premindo com outro uma vez a tecla numérica 6 ou a 7, verificar-se-á que no *écran*, entre o algarismo 3 e a palavra REM da primeira linha do programa, aparece um indicador parecido com este símbolo: >. Este indicador chama-se *cursor do programa*. Decerto já se terá observado que, quando se teclam linhas de um programa e se carrega em ENTER, há sempre um > próximo do número da linha então introduzida.

A primeira coisa a fazer para editar uma linha é mover o cursor do programa até essa linha. A seta apontada para baixo move o cursor do programa para a linha de baixo e a apontada para cima move-o para a linha acima. Como o leitor deseja editar a linha 10, deve manter um dedo carregando na tecla CAPS SHIFT e, com outro dedo, premir a tecla numérica 6 cinco vezes. O > deverá estar agora entre o 10 e o PRINT da quarta linha do programa. Se carregou uma vez a mais na tecla 6, então prima a tecla 7 uma vez, para levar o cursor do programa à posição requerida.

A seguir deve-se copiar a linha 10 para a zona inferior do *écran*, onde pode ser alterada. Olhando para a zona superior esquerda do teclado, ver-se-á a palavra EDIT por cima da tecla 1. Ao manter carregada a tecla CAPS SHIFT, e premindo depois a tecla 1 faz-se uma cópia da linha onde se encontra o cursor do programa. Verifique agora se o cursor do programa está na linha 10. Então, mantendo um dedo na tecla CAPS SHIFT, prima a tecla 1 uma vez. Imediatamente aparece uma cópia da linha 10 na zona inferior do *écran*, semelhante a esta:

```
10 K PRINT "Deram-me Esc.";a
```

Note-se que o cursor intermitente K está mesmo à direita do número 10. Partindo do ponto onde aparece o cursor intermitente, pode-se apagar (DELETE) o que está à esquerda do cursor ou inserir o que for necessário. E se não se quer alterar nada nessa posição do cursor intermitente, pode-se movê-lo para a esquerda ou direita até ao ponto onde se fará a alteração, usando as setas apontadas para a esquerda ou para a direita. Movendo-o para a direita, ele convenientemente mudará de K intermitente a um cursor L intermitente. Como se pretende alterar o ponto e vírgula da linha 10 para uma vírgula, deve mover-se o cursor intermitente para a direita. Mantém-se um dedo sobre a tecla CAPS SHIFT e, com outro dedo, carrega-se na tecla 8 até a linha parecer-se com esta:

```
10 PRINT "Deram-me Esc.",La
```

Agora, usando as teclas CAPS SHIFT e 0 como habitualmente, apaga-se o ponto e vírgula, para que a linha 10 se veja como esta:

```
10 PRINT "Deram-me Esc."La
```

Agora escreve-se uma vírgula (teclas SYMBOL SHIFT e N), devendo ficar assim:

```
10 PRINT "Deram-me Esc.",La
```

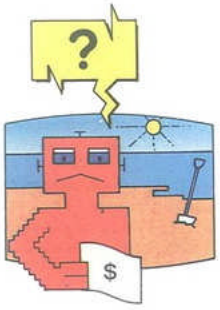
Acabou a edição desta linha. O cursor intermitente ainda se encontra na linha, mas não importa. Prime-se simplesmente ENTER e esta nova linha 10 desaparece do fundo do *écran* e surge na lista da parte superior do *écran*, no lugar da anterior linha 10. Se executar (RUN e ENTER) o programa, dar-se-á conta do que mudou! Pratique agora a edição de linhas, alterando as linhas 20, 30 e 40 da mesma maneira.

As setas, esquerda e direita, também se usam para editar uma linha *antes* de ser introduzida (antes de premir ENTER). Isto poupa muito tempo quando se comete um erro no início da linha, e só se tendo reparado nele depois de digitar quase toda a linha. Nesta situação, há duas opções: ou apaga-se tudo à esquerda até ao ponto onde o erro foi cometido, emenda-se e depois volta-se a digitar aquilo que já estava bem (e perdendo muito tempo) ou se usa a seta esquerda para levar o cursor até ao erro, corrige-se e, em seguida, move-se o cursor para a posição donde partiu, utilizando a seta direita.

Outras variáveis numéricas

Tente agora o leitor escrever programas mais complicados que usem variáveis numéricas. Embora usualmente se vejam programas completamente transcritos, com explicações e tudo, não pense o leitor que um programa surge na mente já todo feito e pronto a funcionar! Não, quase todos os programas começam com umas poucas linhas, às quais o programador vai agregando outras e mais outras.

▲ *Por se ter alterado o ponto e vírgula, na linha 10, para uma vírgula, há agora um espaço vazio entre Esc. e o valor impresso nesta linha.*



Primeiro tem de se decidir que tipo de jogo se irá inventar, ou que tipo de problema se necessita resolver. Precisaríamos, por exemplo, de um programa para planear os gastos nas férias? Ou talvez se queira saber quanto dinheiro há para as férias e quanto se poderá despender por dia. Para começar, deve-se decidir qual a informação a dar ao computador. Neste caso, é necessário saber:

a) quanto dinheiro se possui.
b) quanto dinheiro se receberá para despesas diárias.
c) quanto dinheiro deverá sobrar no fim das férias.
d) quantos dias durarão as férias.

Depois deve-se decidir como introduzir estes dados no computador. Ensaiem-se as linhas seguintes, copiando-as cuidadosamente, escrevendo as aspas tal e qual se mostra a seguir:

```
10  PRINT  ''Quantos Esc.''
20  PRINT  ''Ja tenho''
30  INPUT  a
40  PRINT  a
50  PRINT  ''Quanto se recebera para despesas diarias''
60  INPUT  b
70  PRINT  b
80  PRINT  ''Quanto dinheiro''
90  PRINT  ''devera sobrar''
100 PRINT  ''no fim das ferias?''
110 INPUT  c
120 PRINT  c
130 PRINT  ''Quantos dias''
140 PRINT  ''durarao as ferias''
150 INPUT  d
160 PRINT  d
```

Lembremo-nos de que o comando INPUT está na tecla I. As linhas 40, 70, 120 e 160 não são necessárias na entrada de dados, mas sim para imprimir no *écran* os números escolhidos, de forma a poder-se recordá-los e vê-los. Execute-se o programa escrito até agora, para ver o que faz. Assegure-se que se atinge a última linha e, no caso contrário, compare-se cuidadosamente a lista do programa introduzido com a deste livro, até encontrar o erro. Edite-se depois essa linha como foi explicado anteriormente. Emendem-se todos os erros até o programa ser executado até à última linha. Agora o computador possui toda a informação necessária.

Mas falta dizer-lhe o que fará com essa informação. Querendo saber quanto dinheiro se pode gastar, deve-se instruí-lo para somar o dinheiro que já se tem (a) ao dinheiro que se receberá (b), e depois subtrair de a+b a quantia que deverá sobrar no fim das férias (c). Assim, carregue-se em LIST e ENTER e agregue-se a linha seguinte:

```
170  PRINT  ''Tenho Esc.'';a+b -c;''para gastar''
```

Note-se que existe um ponto e vírgula entre as segundas '' (aspas) e **a+b -c,** e outro ponto e vírgula depois desta expressão e antes das terceiras ''. Podia-se ter usado vírgulas em lugar dos pontos e vírgulas, mas nesse caso ficaria um espaço muito grande, entre Esc. e o resultado impresso pelo computador, quando se executasse o programa. O uso de pontos e vírgulas significa que não haverá espaço nenhum entre eles. No entanto, é necessário haver um espaço entre o resultado e a palavra *para* que está na segunda parte da instrução PRINT. Por isso, deve-se deixar um espaço em branco (premindo uma vez a tecla BREAK SPACE) imediatamente depois das terceiras '' (aspas). Quando se digitarem linhas de programas, será muito importante copiá-las exactamente como estão no livro, não esquecendo de dar os espaços em branco correctamente. Se algum for omitido rapidamente se notará ao executar o programa e, então, deve-se editar a linha errada. De aqui em diante, e para facilitar ao leitor a introdução correcta das linhas, os espaços a usar numa instrução PRINT serão substituídos por um traço, assim −. Um − significa simplesmente que se deve usar um espaço em branco.

É necessário agora agregar mais linhas a este programa. Para calcular quanto dinheiro se pode gastar por dia, deve-se instruir o computador para dividir a quantia total que tem para gastar (a+b -c) pelo número de dias que se vai passar em férias (d). Por isso agregam-se as linhas:

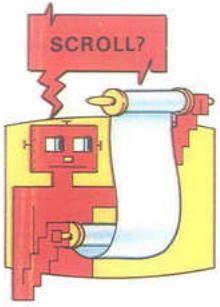
```
180  PRINT  ''Posso−gastar−Esc.'';(a+b -c)/d
190  PRINT  ''por−dia''
```

A linha 190 não necessita de um espaço entre as primeiras '' e a palavra *por*, porque não está a seguir a

(a+b -c)/d, na linha 18Ø; o PRINT da linha 19Ø é executado na linha abaixo. O programa está pronto para ser executado, no entanto não tem linhas que expliquem onde começa e onde acaba. Pode dar-se-lhe um título e um fim com as linhas.

```
  5 REM  "FERIA$S"
200 STOP
```

O comando STOP está a vermelho na tecla A. Agora examine-se cuidadosamente o programa.



? Problema 16: O programa está tão comprido que não se pode vê-lo todo no *écran!* Como ver as primeiras linhas?

☑ Com o comando LIST (tecla K e ENTER). Agora vêem-se as primeiras linhas do programa, até à linha 16Ø. E por baixo delas, sobre o lado esquerdo, vê-se a palavra:

```
scroll?
```

que significa rolo de papel com coisas escritas. Em cada momento, só pode ver-se uma parte do que está escrito. Então, enrolando dum lado e desenrolando do outro, pode ver-se o que segue. O computador está a perguntar se quer que faça o mesmo com as linhas do programa que tem armazenado na memória. Se *premir* qualquer tecla (excepto CAPS SHIFT, SYMBOL SHIFT, BREAK SPACE ou N) o computador 'enrolará' as linhas superiores do *écran*, e 'desenrolará' mais algumas linhas para além do fundo do *écran*, de modo a poder ler-se a parte seguinte do programa. Quando já não há mais linhas para desenrolar, ou 'scroll', o computador imprime mais um dos seus 'OK'. Se *não* se quiser desenrolar *(scroll)* o programa, mas se deseja guardar parte no *écran* para edição de linhas, prime-se a tecla BREAK SPACE. Aparecerá uma mensagem no fundo do *écran* dizendo:

```
D  BREAK-CONT repeats, 0:1
```

Depois pode-se agregar ou editar as linhas na parte do programa que se vê no *écran*.

Voltamos ao programa. É sempre boa ideia explicar em poucas palavras para que serve o programa. Assim, se alguém mais o utilizar, saberá a sua finalidade. Adicionem-se mais estas linhas:

```
  6 PRINT  "Este-programa-ajuda-a"
  7 PRINT  "planear-os-gastos-das-ferias"
```

Execute-se o programa e verifique-se se funciona bem.

? Problema 17: Este programa funciona muito bem. Mas pode melhorar-se a sua apresentação. Deve haver uma linha em branco entre as duas linhas que descrevem a sua utilidade e as linhas onde o computador pede dados. Também deve haver outra linha em branco entre as linhas nas quais o computador pede dados e as linhas de resultados. E ainda deve haver mais uma, entre os dois resultados.

☑ Para deixar uma linha em branco no *écran*, basta introduzir um PRINT sem escrever nenhuma mensagem. Por isso adicione-se:

```
  8 PRINT
  9 PRINT
165 PRINT
175 PRINT
```

Execute-se o programa outra vez. Parece bastante bem. Mas os programadores quase nunca ficam satisfeitos!

? Problema 18: Como se pode tornar este programa ainda mais interessante e útil?

☑ Usando duas instruções IF... THEN. Porquê? Porque, neste momento, se alguém introduzir, por exemplo, 25 em INPUT a (linha 3Ø) e 27 em INPUT b (linha 6Ø), também pode entrar 7Ø em INPUT c (linha 11Ø). E o computador imprimiria, na linha 17Ø:

```
Tenho Esc.-18 para gastar
```

O que seria absurdo! Uma quantia negativa (−18)? Esta situação remedeia-se usando as instruções IF...THEN. Agreguem-se estas duas linhas:

```
124 IF  a+b -c<=Ø THEN PRINT "Que - disparate! - Nao - pode - restar - - tanto - no - fim - das - ferias"
125 IF  a+b -c<=Ø THEN GOTO 8Ø
```

O comando IF encontra-se na tecla U, e o THEN, a vermelho na tecla G. O símbolo <= (é menor ou

▲ *Para melhorar o aspecto dos resultados do programa, pode-se usar a instrução PRINT, por si só, para dar espaços entre linhas do* écran.

▲*Para introduzir estes símbolos, mantém-se um dedo na tecla SYMBOL SHIFT, enquanto se carrega nas teclas E ou Q.*

igual a) está a vermelho na tecla Q (não se pode usar nunca dois símbolos separados < e =, ou > e = para constituir as desigualdades <= ou >=). A linha 124 diz ao computador que, se (IF) o dinheiro que resta no fim das férias for menor ou igual a Ø Esc., então (THEN) ele deve imprimir uma mensagem explicando que quem executou o programa introduziu um número demasiado grande em INPUT c. Em outras palavras, ele ou ela quis ter mais dinheiro no fim das férias do que quando as começou.

A *linha 125* diz ao computador que, se (IF) o dinheiro no fim das férias for menor ou igual a Ø Esc., então (THEN) ele saltará atrás para a linha 8Ø. Por isso quem executar o programa terá outra oportunidade para introduzir um número que faça sentido, em INPUT c.

Depois de introduzir estas duas novas linhas, liste-se (LIST) o programa no *écran*, analisando-o cuidadosamente até ter a certeza que se compreende o que sucede em cada linha. E só então se manda executá-lo (RUN).

Variáveis de texto (variáveis alfanuméricas)

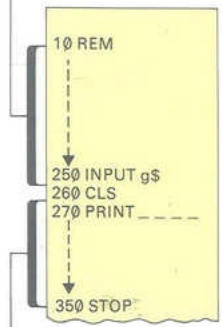
(Repetição das mesmas instruções, sendo os dados palavras diferentes de cada vez.)

Usámos as letras **a, b, c** e **d** como variáveis numéricas, mas poderíamos ter escolhido outras letras quaisquer. Com palavras ou grupos de letras — e outros símbolos: $, &, ?, £, etc., aqui interpretados como letras — variáveis (*variáveis de texto*) faz-se o mesmo, excepto que depois da letra a, por exemplo, é obrigatório usar-se o sinal $ (indica que é uma variável *de texto*). Este sinal é o símbolo vermelho da tecla 4. Apresentamos a seguir um programa para escrever cartões de agradecimento e que usa variáveis de texto. (Na linha 27Ø do programa, deve-se digitar, está claro, o endereço e não 2, Planeta *Basic*.)

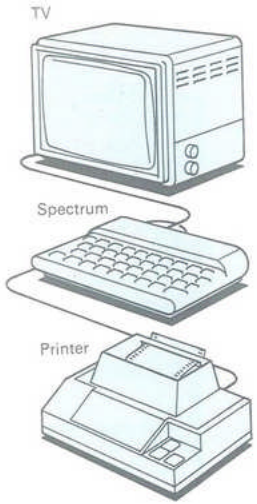

```
1Ø  REM  "OBRIGADO"
2Ø  PRINT  "Este – programa – ajuda – a – escrever"
3Ø  PRINT  "um – bilhete – de – agradecimento."
4Ø  PRINT
5Ø  PRINT
6Ø  PRINT  "Diga – que – presente – recebeu. – Foi – um/a"
7Ø  INPUT  a$
8Ø  PRINT  a$
9Ø  PRINT  "Diga – um – termo – que – descreva – o/a – –"; a$
1ØØ  INPUT  b$
11Ø  PRINT  b$
12Ø  PRINT  "Quem – ofereceu – o/a";a$
13Ø  INPUT  c$
14Ø  PRINT  c$
15Ø  PRINT  "Escolha – um – termo – ou – termos – que – descrevam – o – tempo – que – faz."
16Ø  INPUT  d$
17Ø  PRINT  d$
18Ø  PRINT  "Escolha – um – termo – ou – termos – que – descrevam – o – estado – da – sua – familia."
19Ø  INPUT  e$
2ØØ  PRINT  e$
21Ø  PRINT  "Como – assina – o – seu – nome?"
22Ø  INPUT  f$
23Ø  PRINT  f$
24Ø  PRINT  "Que – data – vai – usar?"
25Ø  INPUT  g$
26Ø  CLS
27Ø  PRINT  "– – – – – – – – – – 2, – Planeta – Basic.»
28Ø  PRINT  "– – – – – – – – – –";g$
29Ø  PRINT
3ØØ  PRINT  "– – Caro/a – amigo/a – ";c$
31Ø  PRINT  "– – – – Os – meus – agradecimentos – pelo/a – ";a$;". – E – realmente –";b$;". – De – facto – foi o/a – melhor – ";a$;" – que – ja – tive."
32Ø  PRINT  "– – – – Espero – que – tenha – tido – bom – tempo. – Por – aqui – o – tempo – tem – sido –";d$;". – Gostara – de – saber – que – a – familia – esta –";e$;". – Agradeço – novamente – pelo/a –";a$;"."
33Ø  PRINT  "– – – – – – Abraços"
34Ø  PRINT  "– – – – – – do –";f$
35Ø  STOP
```

O comando CLS está na tecla V. Como se vê, as

Estas linhas recolhem informação para escrever o bilhete



Estas linhas imprimem o bilhete no *écran*



▲ *Para se obter uma cópia, escrita em papel, do bilhete que está impresso no écran, é preciso ligar ao computador uma impressora.*

linhas até 250 reúnem informações e mostram-nas no *écran*. As linhas 270 até ao fim imprimem o cartão. Na linha 260 há uma instrução muito útil, CLS, que diz ao computador para limpar o *écran*. Sem ela, o bilhete de agradecimento apareceria no fundo do *écran*, juntamente com as perguntas e respostas dadas. Quem dispuser de uma impressora, poderá executar o programa e imprimir o bilhete em papel. E depois poderá executá-lo novamente, introduzindo outras variáveis de texto para imprimir novo bilhete para outro amigo. Mesmo que não se disponha de uma impressora, é divertido executar este programa com diversas variáveis, sejam elas sensatas ou disparatadas! O computador imprimirá o que se lhe der como entrada de dados. Procure o leitor executar o programa várias vezes e depois introduzir modificações que o melhorem.

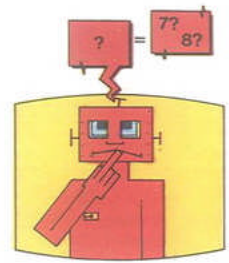
Por exemplo, na linha 90 deixaram-se dois espaços, acertadamente, depois do artigo o/a, de forma a que, quando o termo descritivo do presente fosse adicionado à linha, não se separasse em duas partes, com as duas primeiras letras aparecendo no fim da linha e o resto desse termo na linha seguinte. Mas quando as linhas 150 e 180 aparecem no *écran*, a palavra *descrevam*, em ambos casos, está separada em duas partes. O leitor pode adicionar espaços ou dividir cada uma destas linhas em duas instruções PRINT. Haverá outros sítios onde seja de fazer alterações deste tipo?

Por outro lado, este bilhete diz pouca coisa. Talvez queira incluir um parágrafo no qual mencione um livro que esteja a ler, ou um programa de televisão que tenha visto e dar as suas impressões sobre eles. Para fazer isto, o leitor deve imaginar a forma das frases a usar, tais como 'Tenho visto ultimamente um programa na televisão chamado... os actores são... e... acho este programa muito... diga-me: o que pensa dele?' Depois deve adicionar linhas para introduzir a informação (INPUT) necessária. E agregar as linhas para imprimir (PRINT) as novas frases no bilhete.

Uso conjunto de variáveis numéricas e de texto

Eis um jogo que usa variáveis numéricas e de texto. Recordemos que é preciso introduzir todas as aspas e pontos e vírgulas.

```
10  REM "PALPITES"
20  PRINT "Jogo - de - adivinhas - para - DOIS - adversarios."
30  PRINT
40  PRINT "O - nome - do - PRIMEIRO - jogador - e"
50  INPUT a$
60  PRINT a$
70  PRINT "O - nome - do - SEGUNGO - jogador - e"
80  INPUT b$
90  PRINT b$
100 PRINT
110 PRINT "Agradeço-te, -";a$;" - e - tambem - a - ti -";b$;"."
120 PRINT b$;", - nao - olhes - para - o - ecran."
130 PRINT a$;", - por - favor - tecle - um - numero - entre - 1 - e - 20 - e - depois - carrega - em - ENTER."
140 INPUT a
150 PRINT "Muito -- bem! - Agora, - para - continuar, - carrega - em - ENTER."
160 INPUT z$
170 CLS
180 PRINT "Pede - agora - a";b$;"que - olhe - para - o - ecran."
190 REM Adivinha - ate - acertar - (iteracao)
200 PRINT b$;", - adivinha - qual - foi - o - numero - entre - 1 - e - 20."
210 INPUT b
220 CLS
230 IF b<>a THEN PRINT "Azar! -";b;"- nao - esta - correcto."
240 IF b<>a THEN GOTO 190
250 PRINT "MUITO - BEM, - ";b$;",- ACERTASTE! - O - numero - era -";a
260 PRINT
270 PRINT
280 PRINT "Queres - jogar - outra - vez? - Digita - s - para - sim - ou - n - para - nao."
290 INPUT c$
300 IF c$="s" THEN CLS
310 IF c$="s" THEN GOTO 20
320 IF c$<>"n" THEN GOTO 280
330 CLS
340 STOP
```



< é menor que

> é maior que

<> é menor ou maior que

▲ *Se o palpite do jogador foi superior ou inferior à resposta correcta, o computador imprimirá uma mensagem de 'Azar!' e dará ao jogador outra oportunidade.*

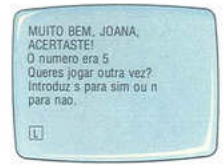
O símbolo <> está a vermelho na tecla W. A primeira parte deste programa é muito fácil de seguir.
A linha 20 explica que é um jogo de adivinhas.
As linhas 40 até 80 introduzem, no computador, os nomes dos dois jogadores para as variáveis de texto a$ e b$.
Na linha 140, o primeiro jogador introduz um número para a variável numérica **a** (equivale a dizer que se guardou um número numa posição da memória chamada '**a**').
Na linha 210, o segundo jogador introduz o seu palpite para a variável numérica **b** (guarda-o numa posição da memória chamada '**b**').

Agora torna-se um pouco mais difícil seguir o programa. Se o segundo jogador adivinhou, então é porque o número na posição **b** é igual ao número na posição **a**. Mas se não adivinhou? Se o segundo jogador não adivinhou, então b é menor ou maior que **a**. O símbolo < significa *é menor que*, e o > significa *é maior que*, e o leitor pode usar o símbolo <> para significar *é diferente de* (nunca use os símbolos isolados < e > para, juntando os dois, constituir <>. Use sempre o símbolo na tecla W). E continuando a análise do programa:
Na linha 230 diz-se ao computador que se o segundo jogador falhou (IF b<>a), então (THEN) deve imprimir a mensagem 'Azar! etc.'.
Na linha 240 diz-se ao computador que se o segundo jogador falhou (IF b<>a), então (THEN) deve ir directamente (GOTO) para trás, à linha 190, dando ao segundo jogador nova oportunidade para adivinhar o número. Assim, as linhas de 190 a 240 formam um ciclo *(loop)*, e o computador repete essas linhas até o segundo jogador adivinhar o número, em INPUT **b**. Para tornar o ciclo tão claro como possível, escreveu-se a linha 190 com um REM, indicativo de onde começa o ciclo. O comentário depois do comando REM, explicando o que se faz nessa parte do programa, quando usado dentro do programa, não necessita estar entre aspas (NÃO é o título do programa) e pode ter espaços em branco entre palavras e ser tão comprido quanto se queira.

Quando se adivinhar o número, isto é, quando b=a, o computador não tomará em conta as linhas 230 e 240, e irá directamente à linha 250, onde é instruído para imprimir a mensagem 'Muito bem, etc.', seguindo depois até à linha 280. Aqui encontra mais algumas instruções IF... THEN. Se (IF) alguém introduz um s, então (THEN) o computador limpa o *écran* (CLS) e vai para a linha 20, onde começa um novo jogo.

? Problema 19: Mas se alguém introduzir outra letra, diferente de s?
✓ Se for a letra n, o computador ignorará as linhas 300, 310 e 320 e continuará na linha 330, onde limpará o *écran* e parará (STOP). Se a letra for outra qualquer, porque por engano o jogador não teclou nem s nem n, ou se escreveu S ou N, então o computador ignorará as linhas 300 e 310. Mas na linha 320 saltará atrás à linha 280, dando ao jogador outra oportunidade para premir s ou n, e só estas.

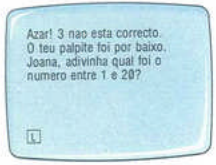
? Problema 20: Como se pode parar a execução dum programa, sem sair dele?
✓ As linhas 150 e 160 tratam desse problema. A execução do programa pode parar quanto tempo se quiser, usando uma linha de entrada (INPUT) de dados. Havendo uma linha de entrada (INPUT) de dados, a execução do programa pára, até se teclar o dado e premir-se ENTER, ou simplesmente carregando em ENTER. Só depois continuará a execução. Assim, a linha 150 diz ao jogador que, para continuar o jogo, deverá premir a tecla ENTER. Carregando na tecla ENTER faz-se a entrada de dados que o computador espera na linha 160, e depois continua a execução do programa.



▲ *Quando o palpite do jogador é igual à resposta certa, o computador imprime uma mensagem de 'Muito bem' e oferece outro jogo.*

Para melhorar "PALPITES"

1) Veja se pode imaginar algumas formas de melhorar o *aspecto* do programa no *écran*. Por exemplo, mais algumas linhas PRINT ajudariam.
2) Nesta altura, alguma coisa não está bem, não só com o aspecto do jogo mas também com a maneira como está programado (a sua *estrutura*). Ao adivinhar, não se tem qualquer ideia se, na tentativa anterior, o número ensaiado foi maior ou menor que o correcto. O jogo seria mais interessante se o leitor agregasse uma linha instruindo o computador a imprimir uma mensagem dizendo «O teu palpite foi por baixo» ou «O teu palpite foi por cima». Para isso,



▲*Com as instruções IF ... THEN, o computador dá sugestões ao jogador quanto à resposta certa.*

podem usar-se mais instruções IF... THEN. Ensaie estas:

```
233 IF b<a THEN PRINT "O - teu - palpite - foi - por - baixo."
235 IF b>a THEN PRINT "O - teu - palpite - foi - por - cima."
```

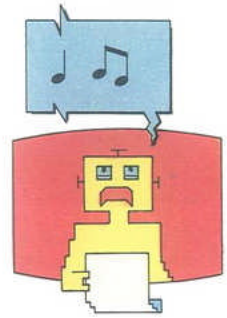
O símbolo < (é menor que) está a vermelho na tecla R e o > (é maior que) está na tecla T. Se gostar deste jogo e quiser gravá-lo (SAVE) em *cassette*, para poder jogar sem ter de voltar a teclá-lo, veja a pág. 65.

Revisão sumária

Neste capítulo aprendeu:

1) como usar o computador em matemática elementar.
2) como usar variáveis numéricas e de texto.
3) como usar os pontos e vírgulas para que os programas sejam atraentes no *écran*, quando são executados (RUN).
4) como apagar linhas do programa depois de introduzidas e como agregar outras novas.
5) como usar REM para dar um título ao programa.
6) como editar programas, usando a tecla CAPS SHIFT junto com as setas por cima das teclas 5, 6, 7 e 8, e também com EDIT por cima da tecla 1.
7) como desenrolar *(scroll)* um programa comprido para ser analisado por partes, no *écran*.
8) como deixar linhas em branco no *écran* usando PRINT.
9) para que servem e como se usam as instruções IF... THEN.
10) como limpar o *écran* com CLS.
11) como facilitar a compreensão do que se passa quando há um ciclo *(loop)* num programa, introduzindo uma linha REM no início do ciclo.
12) o modo de usar a tecla ENTER numa entrada de dados (INPUT).

3 Obtenção de som



O *ZX Spectrum* pode ser programado para emitir sons estranhos ou para tocar música!
Quando se introduz som num programa, uma das novas linhas talvez se pareça com esta:

```
10 BEEP 1,0
```

10 é, evidentemente, o número da linha.
BEEP é o comando, ou instrução, utilizado para emitir som.

1,: O primeiro número do par 1,0 dá o tempo de duração do som, breve ou mais demorado, conforme se usa números de 1 a 10. Neste exemplo, o número 1 diz ao computador que o som deve durar um segundo. Se fosse o número 2, o som obtido duraria 2 segundos, com 3 duraria 3 segundos e assim sucessivamente. Mesmo que o leitor não trabalhe habitualmente com números decimais, saberá certamente que 0,5 equivale a uma metade. Assim 1.5 produzirá um som com a duração de 1 segundo e meio, com 2.5 durará 2 segundos e meio, etc. (Os leitores que usam frequentemente números decimais gostarão de saber que, para especificar a duração do som, poderão utilizar todos os números entre 0.01 e 10.49.)
,0: O segundo número do par 1,0 serve para escolher o som (nota) e pode aumentar desde 0 até 69 ou decrescer até -60. Neste exemplo, o número 0 diz ao computador que toque a nota dó (natural). O valor mais alto a usar para obter sons harmoniosos é cerca de 50 e o mais baixo à volta de -18.

Obtenção de efeitos especiais

Quando se pretende que o computador produza ruídos estranhos, tais como explosões, disparos ou o vazio do espaço sideral, ter-se-á de usar notas entre 51 e 69 ou entre -19 e -60. Ensaie-se este programa:

```
10 REM "NOITE"
20 BEEP 1, -40
30 BEEP 1, -30
```

```
40 BEEP 1,67
50 BEEP 1, -50
60 BEEP 1, -30
70 BEEP 1, -40
80 STOP
```

Notar que BEEP está a vermelho, por baixo da tecla Z, de modo que é necessário passar ao modo E e depois premir Z enquanto um outro dedo carrega na tecla CAPS SHIFT. Para repetir o som produzido depois de correr o programa, bastará premir RUN outra vez.

Composição de melodias

Para compor melodias, devem-se usar notas entre 50 e −18. Ensaiemos esta:

```
10 REM "SOLDADO"
20 BEEP .5,7
30 BEEP .5,4
40 BEEP .5,0
50 BEEP .5,9
60 BEEP 1,7
70 BEEP .5,5
80 BEEP .5,2
90 BEEP .5, -1
100 BEEP .5, -4
110 BEEP 1,0
120 BEEP .5,7
130 BEEP .5,4
140 BEEP .5,0
150 BEEP .5,9
160 BEEP 1,7
170 BEEP .5,5
180 BEEP .5,7
190 BEEP .5,9
200 BEEP .5,11
210 BEEP 1,12
220 STOP
```

Uso repetido de um grupo de linhas

O leitor talvez tenha notado que, no programa "SOLDADO", as linhas 20 a 70 são exactamente iguais às linhas de 120 a 170. Sempre que, ao escrever um programa, haja um grupo de linhas que devem ser usadas mais do que uma vez, será uma boa ideia *escrevê-las uma só vez*, depois da linha STOP, e usar uma instrução especial que as mande executar cada vez que se queira. Essa instrução especial é GOSUB (na tecla H), seguida pelo número da primeira linha do grupo das linhas que se querem executar. Para exemplificar, o leitor deve escrever de novo o programa "SOLDADO", usando GOSUB em lugar das linhas 20 a 70 e também das linhas 120 a 170. O programa ficará assim:

```
10 REM "SOLDADO"
20 GOSUB 1000
30 BEEP .5,2
40 BEEP .5, -1
50 BEEP .5, -4
60 BEEP 1,0
70 GOSUB 1000
80 BEEP .5,7
90 BEEP .5,9
100 BEEP .5,11
110 BEEP 1,12
120 STOP
```

Agora deve-se introduzir o grupo das linhas que GOSUB invoca, depois da linha STOP. Para salientá-lo bem no programa, é boa ideia dar a estas linhas um número muito maior do que os usados no programa. Esta a razão para escrever GOSUB 1000 e não GOSUB 130. Também será vantajoso que a primeira linha do grupo seja um REM, explicando para que servem as linhas seguintes. A última linha do grupo deve ser, obrigatoriamente, o comando RETURN (na tecla Y). A função do comando RETURN é dizer ao computador que volte ao programa principal. Assim, o leitor deve agregar as linhas:

```
1000 REM Seis notas
1010 BEEP .5,7
1020 BEEP .5,4
1030 BEEP .5,0
1040 BEEP .5,9
1050 BEEP 1,7
1060 BEEP .5,5
1070 RETURN
```

▲*Quando se quer usar mais do que uma vez um grupo de linhas no mesmo programa, devem-se introduzir no fim do programa, como um grupo separado. E usa-se a instrução GOSUB, seguida do número da primeira instrução do grupo de linhas, para que o computador as execute todas as vezes que forem necessárias.*

▲*Usando efeitos de som, os programas tornam-se muito mais interessantes. Deve-se experimentar tantos sons diferentes quantos se possa imaginar.*

Neste exemplo só se economizaram duas linhas. Mas se houvesse mais linhas em GOSUB, ou se fosse usada mais do que duas vezes, ter-se-ia poupado bastante espaço. Mais importante é o facto de o uso do GOSUB tornar um programa muito mais fácil de seguir e compreender. Quando é fácil seguir um programa, diz-se que está 'bem organizado' ou, melhor ainda, 'bem estruturado'. Uma boa estrutura define um bom programador!

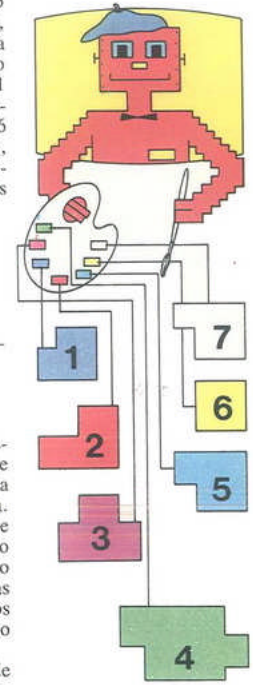
Tente agora o leitor compor melodias suas ou fazer um programa para tocar uma melodia conhecida. Encontrará ajuda valiosa no capítulo sobre som do livro *ZX Spectrum BASIC Programming*. Pode-se também experimentar com efeitos especiais. Conseguirá o computador emitir um som parecido com o de uma chaleira com água a ferver? Ou o de uma porta a ranger?

Revisão sumária

O leitor aprendeu como programar músicas e efeitos especiais usando a instrução BEEP. Também aprendeu como escrever programas mais fáceis de seguir, e às vezes mais curtos, usando o comando GOSUB.

4 *Palavras e planos de fundo coloridos*

Em princípio o computador *ZX Spectrum* imprime texto (palavras) a negro sobre um *écran* branco. Mas é fácil aprender a colorir tanto as palavras como o *écran*. Este está dividido em duas zonas. É fácil colorir a margem exterior ou enquadramento à volta do *écran*, também chamada *fronteira*. Basta premir o comando BORDER, que se encontra na tecla B, seguido de um número. Não é difícil saber o número a escolher, porque algumas das teclas numéricas têm o nome das cores por cima. Assim, verifica-se que um 1 dá azul-escuro, 2 dá vermelho, 3 dá magenta (púrpura-claro), 4 dá verde, 5 dá celeste (um azul-claro), 6 dá amarelo, 7 dá branco e Ø dá negro. Deste modo, BORDER 1 dará ao *écran* um enquadramento azul-escuro. Vamos escrever esta instrução como uma das primeiras linhas de um programa:




```
1Ø  REM  "TEXTO COLORIDO"
2Ø  BORDER  1
3Ø  PRINT  "Agora – ve – uma – margem – azul – escura."
```

Ao executar o programa, vê-se uma margem azul-escura. A zona dentro do enquadramento chama-se fundo. É nesta zona que aparecem os programas. Na zona do fundo a cor do texto chama-se a cor da tinta. A cor da tinta muda-se com o comando INK, que se encontra a vermelho a baixo da tecla X. (Relendo o capítulo 1, refrescar-se-á a memória e saber-se-á como obter os comandos a vermelho *a baixo* das teclas alfabéticas.) Depois do comando INK, usa-se um dos números do teclado, como se fez depois do comando BORDER.

Além de se escolher a cor da tinta, também se pode escolher a cor do fundo, imprimindo, por exemplo, 'tinta' vermelha sobre fundo amarelo. Para mudar a

▲O ZX Spectrum é capaz de colorir uma margem, a toda a volta do écran.

▲Também é capaz de imprimir letras e números coloridos num plano de fundo com cor diferente.

cor do fundo debaixo de cada letra impressa, usa-se o comando PAPER (a vermelho debaixo da letra C), seguido de um dos números 1, 2, etc., como em BORDER. Assim, INK 2 imprime tinta vermelha e PAPER 6 muda o fundo de cada letra impressa para amarelo. Adicionando estas linhas ao programa, notar-se-á que, antes de carregar em ENTER, elas estão a branco na parte inferior da margem azul.

```
 40  PRINT  "Prima - ENTER - para - continuar."
 50  INPUT  z$
 60  INK  2
 70  PAPER  6
 80  PRINT  "Agora - tem - letras - vermelhas sobre - um - fundo - amarelo."
```

Se se quiser toda a zona do fundo em amarelo, e não somente a faixa onde se está a imprimir, deve-se colocar uma instrução CLS logo a seguir à instrução PAPER. Como já se sabe, a instrução CLS limpa o *écran*. Mas há duas coisas interessantes acerca de CLS que o leitor não notou antes porque as cores do fundo e da margem eram brancas. A primeira é que só limpa a zona do fundo. A segunda é que, ao limpar a zona total do fundo, pinta-a da cor que tenha sido definida. Se se agregarem as linhas seguintes ao programa, mal se introduz a primeira linha verifica-se que toda a zona do fundo se tornou amarela e que todas as linhas são impressas a vermelho. Não é motivo de preocupação!

```
 90  PRINT
100  PRINT  "Prima - ENTER - para - continuar."
110  INPUT  z$
120  CLS
130  PRINT  "Agora - esta - amarela - toda a - zona - do - fundo."
```

Para se voltar às cores iniciais, é necessário mudar as zonas da margem e do fundo para branco, e a tinta para negro. Por isso introduzem-se mais estas linhas:


```
140  PRINT
150  PRINT  "Prima - ENTER - para - continuar."
160  INPUT  z$
170  PAPER  7
180  CLS
190  BORDER  7
200  INK  0
210  PRINT  "Normal - novamente."
```

Quando se executa o programa, a primeira parte parecerá diferente. Não tem importância; deve-se executá-lo novamente. Verifica-se que a primeira parte do programa será, então, a mesma que anteriormente.

Uso de letras com cores diferentes na mesma linha

Além de mudar a cor de *todo* o texto que está no *écran*, também se pode dar instruções que mudem a cor da tinta de uma dada instrução PRINT, e até só de uma parte da instrução. Juntem-se as seguintes linhas ao programa:

```
220  PRINT
230  PRINT  "Prima - ENTER - para - continuar."
240  INPUT  z$
250  CLS
260  PRINT  INK 2;"A - minha - casa - e - encarnada,"; INK 1;" - mas - tem - o - tecto - azul."
270  PRINT  INK 4;"A - relva - e - verde,"; INK 3;" - e - nao - magenta!!!"
```

Quando se executa o programa, vê-se as letras mudarem de cor (depois de cada instrução INK). Recordemos que, numa instrução PRINT, as diversas partes devem estar separadas por ponto e vírgula ou por vírgula.

▲Pode-se colorir toda a zona central do écran.

▲E imprimir partes diferentes duma frase com mais do que uma cor.

▲ *Também é possível fazer acender e apagar, intermitentemente, as cores do plano de fundo do texto.*

▲ *Ou fazer o branco normal, bastante baço, do écran, contrastar, intermitentemente, com um branco brilhante.*

Letras intermitentes

Para obter letras intermitentes, usa-se o comando FLASH (a vermelho debaixo da tecla V), seguido do número 1. Agreguem-se mais estas linhas:

```
28Ø  PRINT
29Ø  PRINT  FLASH 1;"Oh! – ja – sabia...!"
```

Ou, se além de intermitente, a queremos a encarnado, edite-se a linha 29Ø para:

```
29Ø  PRINT  INK 2; FLASH 1;"Oh! – ja – sabia...!"
```

Pode-se anular a instrução FLASH com outro FLASH, mas seguido pelo número Ø. Eis uma experiência:

```
3ØØ  PRINT
31Ø  PRINT  INK 3; FLASH 1;"PARA – COM ISSO";FLASH Ø;" DOIDO!"
```

E pode-se tornar o plano de fundo mais brilhante, com o comando BRIGHT (a vermelho debaixo da tecla B), seguido pelo número 1. Juntemos estas linhas:

```
32Ø  PRINT
33Ø  PRINT  INK 2;FLASH 1;BRIGHT 1;"EU DISSE – QUE – PARASSE!"
```

Enquanto a frase 'EU DISSE QUE PARASSE!' acende e apaga no *écran*, verifica-se que o branco está muito mais brilhante do que o branco normal, bastante baço, que habitualmente se vê no *écran*. Para anular o brilho intenso, usa-se, à semelhança de FLASH Ø, a instrução BRIGHT Ø.

Um modo diferente para colorir textos

É possível colorir o texto digitando-o directamente a cores nas linhas do programa. Eis como fazê-lo. Se na linha:

```
34Ø  PRINT  "Ja – estou – parado!!!"
```

desenhamos a palavra 'Ja' em cor azul-escura, 'estou' a vermelho, 'parado' a verde e '!!!' a celeste, escreve-se primeiro no teclado:

```
34Ø  PRINT  "
```

como habitualmente. Agora passa-se ao modo [E] (carregando na tecla SYMBOL SHIFT com um dedo, enquanto outro dedo prime a tecla CAPS SHIFT). Depois carrega-se na tecla numérica 1 (para azul-escuro), enquanto se prime com outro dedo na tecla CAPS SHIFT. O cursor intermitente [E] converteu-se num cursor intermitente azul [L]. E agora, escrevendo-se:

```
Ja
```

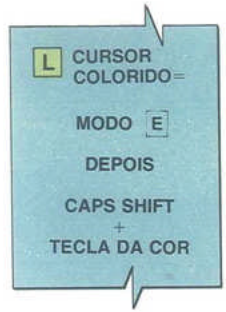
ver-se-á que apareceu escrito a azul. Agora dá-se um espaço, passa-se de novo ao modo [E] e prime-se a tecla numérica 2 (tecla para vermelho), mantendo carregada a tecla CAPS SHIFT com outro dedo. Obtém-se um cursor intermitente vermelho [L]; se escrevermos:

```
estou
```

esta palavra aparecerá a vermelho no *écran*. Da mesma maneira, pode-se escrever 'parado' com um cursor intermitente verde [L] e '!!!' com um cursor intermitente celeste [L]. Quando se atingem as aspas finais da instrução PRINT, é preciso ter a certeza de que aparecem no *écran* a negro passando ao modo [E], e depois obtém-se um cursor intermitente *negro* [L]. Para isso deve-se usar a tecla numérica Ø com CAPS SHIFT. Se as primeiras e segundas aspas *não* estiverem ambas a negro, resultará numa confusão. Porque embora o que aparece no *écran* não mude ao executar o programa, poderá alterar-se a cor das linhas quando se liste (LIST) o programa. Por exemplo, veja-se o que acontece à listagem do programa se se mudarem as aspas finais da linha 29Ø para vermelho! Agora, e para finalizar o programa acerca de cores, adicionaremos a linha:

```
35Ø  STOP
```

Execute-se o programa várias vezes, comparando com cuidado o que acontece no *écran* com a lista de instruções deste livro. Depois, devem-se fazer experiências por conta própria com texto e planos de fundo

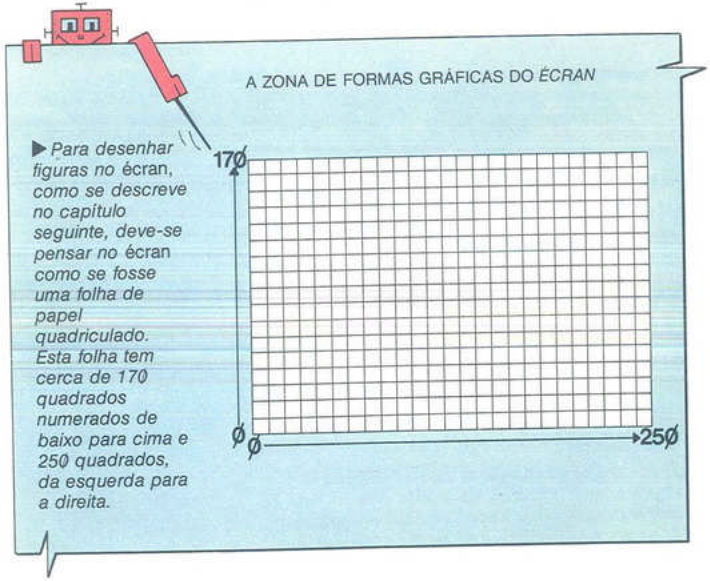


▲ *Teclam-se letras de cores diferentes, directamente, no programa, se se passar dum cursor [L] a outro cursor [L], colorido.*

a cores. O emprego de cores torna sempre os programas mais alegres e emocionantes; regressemos, pois, ao programa "PALPITES" para colorir algumas mensagens. Se se gravou a versão final desse programa numa *cassette*, basta lê-la (LOAD) para a memória do computador (ver pág. 67), e depois é só começar a trabalhar.

Revisão sumária

Neste capítulo, ensinámos a colorir palavras (texto) e planos de fundo usando os comandos:
BORDER, PAPER e INK.
Também ensinámos a conseguir letras intermitentes e a tornar o *écran* mais brilhante com os comandos:
FLASH e BRIGHT.
Ainda ensinámos como introduzir cor directamente nos programas, escrevendo mensagens coloridas entre aspas a negro nas instruções PRINT dos programas.



▶ *Para desenhar figuras no* écran, *como se descreve no capítulo seguinte, deve-se pensar no* écran *como se fosse uma folha de papel quadriculado. Esta folha tem cerca de 17Ø quadrados numerados de baixo para cima e 25Ø quadrados, da esquerda para a direita.*

5 Desenhar com o computador

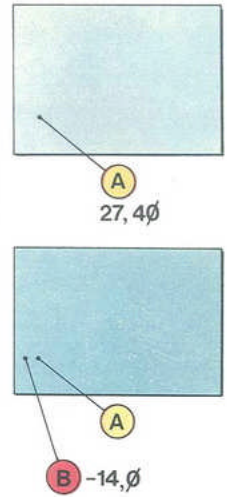
Para desenhar no *écran* usa-se a zona de fundo como prancheta de desenho. Na página anterior vemos um diagrama da zona do fundo no *écran*. Deve-se atribuir-lhe mentalmente uma largura com cerca de 25Ø unidades, e cerca de 17Ø unidades de altura. Querendo definir um ponto num *écran* limpo, é preciso primeiro dizer a que distância está do *lado esquerdo* da zona do fundo (por exemplo, no primeiro diagrama da direita, o *ponto A* está a 27 unidades contadas a partir do lado esquerdo do *écran*), e depois diz-se a distância que vai desse ponto até ao *lado inferior* do fundo (o *ponto A* está a 4Ø unidades do lado inferior). A operação anterior chama-se «atribuir as coordenadas». Será a partir desta origem que se desenhará. A instrução PLOT, seguida pelas coordenadas separadas por uma vírgula, permite dizer ao computador onde irá começar o desenho. Supondo que o desenho vai começar no ponto A, escreve-se no computador a linha:

```
PLOT  27,4Ø
```

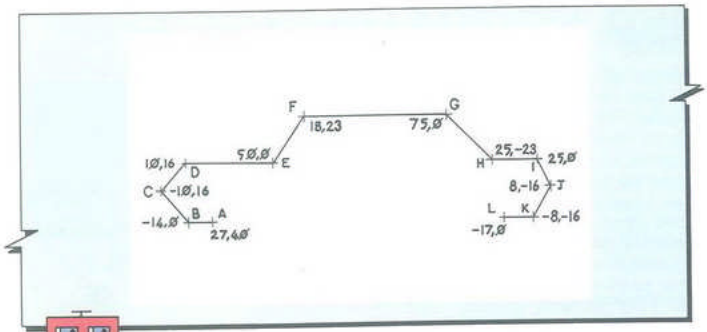
O comando PLOT está na tecla Q. Quando se carrega em ENTER, aparecerá um pequeno ponto no *écran*. Para começar o desenho usa-se a instrução DRAW, seguida pelas coordenadas de outro ponto do *écran*. Não se deve esquecer que a origem está no ponto A. Portanto as coordenadas a usar em DRAW serão em relação ao ponto A: a primeira das coordenadas dirá ao computador se o utilizador quer ir para a *direita* ou para a *esquerda* do ponto A e a que distância, e a segunda deve dizer a que distância irá mas *para cima* ou *para baixo* do ponto A. Suponhamos que se pretende uma linha do ponto A para o ponto B, que como se mostra no segundo diagrama, está a 14 unidades à esquerda do ponto A e à mesma altura (nenhumas, ou zero, unidades para cima ou para baixo). Se for para a esquerda ou para baixo, usa-se um sinal menos (negativo). Depois traça-se a linha para o ponto B:

```
DRAW  -14,Ø
```

O comando DRAW está na tecla W. Quando se



▲ *Usam-se os números dos quadrados para dar ao computador as coordenadas para desenhar os segmentos de recta. Começa-se de um ponto específico, tal como o ponto A (de cima), e então vai-se contando de A para o próximo ponto (ponto B).*



▲ *Eis todas as coordenadas de que se necessita para fazer o computador desenhar a silhueta de um carro.*

carrega em ENTER, o computador traça uma linha do ponto onde se encontrava antes (ponto A) para um novo ponto (ponto B). E, desejando-se unir o ponto B a um outro ponto do *écran*, dão-se as coordenadas desse ponto em relação a B, porque o computador já está em B. Acaba de ser desenhado o primeiro traço do desenho esquemático de um carro. Na página seguinte está um desenho com as primeiras onze linhas, com coordenadas de todos os pontos importantes, começando em A e terminando em L. É importante salientar que as coordenadas dos pontos do desenho estão sempre dadas em relação aos que os antecedem. Introduzam-se estas linhas:

```
1Ø  REM  "CARRO"
2Ø  REM  A carrocaria
3Ø  PLOT  27,4Ø
4Ø  DRAW  -14,Ø
5Ø  DRAW  -1Ø,16
6Ø  DRAW  1Ø,16
7Ø  DRAW  5Ø,Ø
8Ø  DRAW  18,23
9Ø  DRAW  75,Ø
1ØØ  DRAW  25,-23
11Ø  DRAW  25,Ø
12Ø  DRAW  8,-16
```

▼*São necessárias duas semicircunferências sobre as rodas, na base do carro.*


```
13Ø  DRAW  -8,-16
14Ø  DRAW  -17,Ø
```

Executando o programa, verifica-se se o desenho está parecido. Até aqui, tudo foi fácil. Mas vamos que agora queremos fazer um desenho mais complicado. Entre os pontos L e A, queríamos que aparecesse qualquer coisa como o diagrama a baixo.

M está a 37 unidades à esquerda de L. N está a 88 unidades à esquerda de M e A está 37 unidades à esquerda de N. Desde que se alcance M, é fácil traçar o segmento de recta entre M e N. Mas entre L e M queremos traçar uma semicircunferência, e quando se alcançar N é preciso traçar nova semicircunferência entre N e A.

Como desenhar semicircunferências

Será mais fácil ver como se desenham semicircunferências depois de limpar o *écran* (CLS e ENTER). Isto não significa que se percam as linhas já introduzidas (continuarão na memória do computador até suceder uma de três coisas: ou se introduzem novas linhas começando com números iguais ou se escreve NEW seguido de ENTER, ou se corta a corrente de alimentação ao computador). Agora define-se a origem de um novo desenho no *écran*, escrevendo:

```
PLOT  5Ø,1ØØ
```

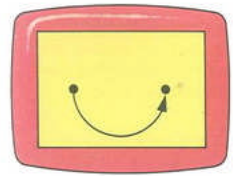
e carrega-se em ENTER. Se queremos desenhar uma semicircunferência começando neste ponto e terminando noutro ponto 15Ø unidades para a direita e Ø unidades para cima, a instrução começa como se fosse traçar um segmento de recta entre dois pontos, como segue (mas não carregue em ENTER):

```
DRAW  15Ø,Ø
```

Adiciona-se uma vírgula, e o comando PI (a verde e por cima de M). Assim:

```
DRAW  15Ø,Ø,PI
```

e prime-se ENTER. Vê-se o computador desenhar uma semicircunferência, do primeiro ponto para o



▲*Usa-se a instrução PI para desenhar uma semicircunferência no sentido contrário ao dos ponteiros dum relógio, de um ponto para outro.*

▲*Quando tiver desenhado a base do carro, a silhueta deverá parecer-se a esta.*

▲*Para acrescentar as janelas, será preciso usar a instrução PLOT para mover-se para uma nova posição no desenho.*

segundo e movendo-se o traço no sentido *contrário ao dos ponteiros do relógio*. Se queremos que o traço se mova no sentido *dos ponteiros do relógio* deve-se usar -PI, em lugar de PI. Agora, voltando ao programa 'CARRO', prime-se CLS e ENTER, seguido de LIST e ENTER.

Para unir L e M, é necessária uma semicircunferência que se mova no sentido *contrário ao dos ponteiros do relógio* para um ponto a 37 unidades para a esquerda, e zero unidades para cima ou para baixo. Por isso, escreve-se no teclado:

```
150  DRAW  -37,0,PI
```

As duas linhas seguintes, agora, são fáceis:

```
160  DRAW  -88,0
170  DRAW  -37,0,PI
```

Assim se regressou ao ponto de partida, o ponto A. Para desenhar as janelas do carro, são necessários dois traços mais e, em ambos os casos, haverá que partir (definir) duma nova origem. Para isso usa-se a instrução PLOT. Agregam-se as linhas seguintes ao seu programa:

```
180  REM  As janelas
190  PLOT  63,72
200  DRAW  118,0
210  PLOT  122,72
220  DRAW  0,23
```

As linhas 190 e 200 traçam um segmento de recta de E a H. O cálculo das coordenadas de E faz-se a partir do ponto A (27,40), somando todos os números e *subtraindo* os números com sinal menos - até chegar a E. Assim, para o primeiro número de PLOT 63,72 ter-se-á 27 -14 -10+10+50 que é igual a 63, e para o segundo número ter-se-á 40+0+16+16+0=72. Para calcular as coordenadas da instrução que traça a linha para H, somam-se todos os números e subtrai-se os números negativos necessários para ir de E até H. Para o primeiro, ter-se-á 18+75+25=118, e para o segundo 23+0 -23=0. As linhas 210 e 220 traçam um segmento que vai desde metade da distância entre E e H até à metade da distância entre F e G. Depois de calcularmos os números 122,72 (da outra instrução PLOT) e 0,23 (da outra instrução DRAW), haverá que desenhar as rodas do carro. O que obriga a que o computador saiba traçar circunferências.

E agora toda a circunferência

Depois de limpar o *écran* (CLS e ENTER), e para desenhar uma circunferência, primeiro escreve-se o comando CIRCLE, que está a vermelho debaixo da tecla H. Assim a instrução começa com:

```
CIRCLE
```

e, a seguir, escrevem-se as coordenadas do *centro* da circunferência, como se desse as coordenadas para uma instrução PLOT. Se o leitor quiser o centro da circunferência no ponto 80,80, a linha ler-se-á:

```
CIRCLE  80,80
```

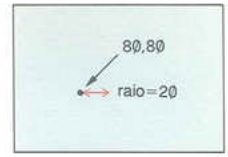
Agora decide-se qual vai ser o *raio* da circunferência. Suponhamos que se escolheu um raio de 20. A linha ficará assim:

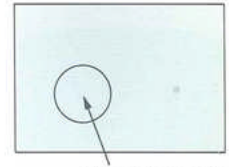
```
CIRCLE  80,80,20
```

Tendo escrito a linha, carrega-se em ENTER e vê-se uma circunferência no *écran*. Volta-se ao programa e adicionam-se estas linhas:

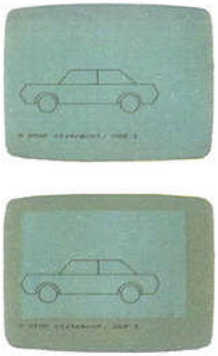
```
230  REM  As rodas
240  CIRCLE  45.5,40,14
250  CIRCLE  170.5,40,14
260  STOP
```

A linha 240 desenha a roda da esquerda. O seu centro foi calculado a metade da distância entre A e N. As coordenadas de A são 27,40 e as de N estão 37 para a direita. Por isso, soma-se metade de 37, igual a 18.5, ao número 27. O que dá 45.5 para o primeiro número. E o segundo número é o mesmo do que o de A, porque o ponto está à mesma altura de A. O raio, 14, foi calculado depois de várias tentativas até que uma delas pareceu mesmo bem. A linha 250 desenha a roda da direita. Calculam-se os números 170.5,40,14 que foram usados para desenhá-la. (Basta calcular as





▲*Para desenhar uma circunferência, dão-se primeiro as coordenadas do centro (tais como 80,80). Depois diz-se ao computador de que raio se quer a circunferência.*



▲*Completado o desenho, pode-se fazer o computador traçá-lo a cores. Também se pode adicionar uma margem colorida.*

coordenadas de M, e lembrar que o centro da roda está a metade da distância entre M e L.)

Linhas coloridas

O carro já está desenhado. Tentemos agora traçar as linhas a cores, usando instruções INK. O carro pode ser encarnado com rodas negras. Experimentemos:

```
 25  INK  2
235  INK  Ø
```

Assim todas as linhas depois da 25 serão desenhadas a encarnado. E depois da linha 235, as linhas serão negras outra vez. Também se pode muito facilmente colorir a margem com:

```
24  BORDER  6
```

Como reduzir o programa

Temos agora um programa com um total de 29 linhas, muitas delas curtíssimas. Poderemos escrever várias numa linha com um só número? Podemos, desde que as separemos pelo símbolo : (dois pontos), que está a vermelho na tecla Z. Podemos fazê-lo de maneira a que o programa se compreenda ainda mais facilmente. Destacando, por exemplo, a mudança de cor das linhas e dando-lhe uma linha própria. Lembremo-nos também de que o computador ignora tudo o que está depois de um REM, por isso não interessa haver uma instrução REM para que o computador lhe obedeça! Podemos reordenar o programa "CARRO" da forma seguinte:

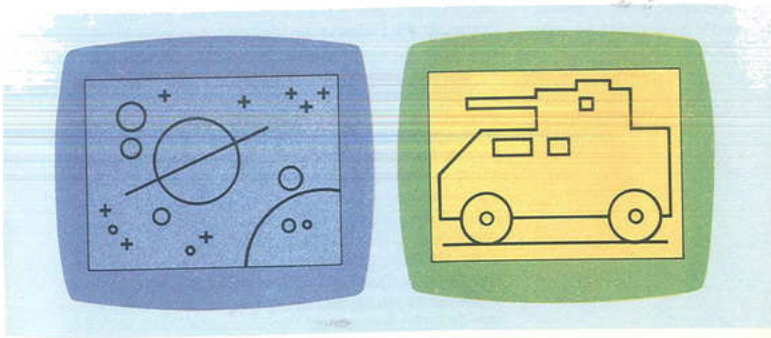
```
 1Ø  REM  "O CARRO": REM A carrocaria
 2Ø  BORDER  6
 3Ø  INK  2
 4Ø  PLOT  27,4Ø: DRAW  -14,Ø: DRAW
-1Ø,16: DRAW 1Ø,16
 5Ø  DRAW 5Ø,Ø: DRAW 18,23: DRAW 75,Ø:
DRAW 25, -23
 6Ø  DRAW  25,Ø: DRAW  8, -16: DRAW
-8, -16: DRAW  -17,Ø
 7Ø  DRAW  -37,Ø,PI: DRAW  -88,Ø: DRAW
-37,Ø,PI
 8Ø  REM  As janelas
 9Ø  PLOT  63,72: DRAW 118,Ø
1ØØ  PLOT  122,72: DRAW Ø,23
11Ø  REM  As rodas
12Ø  INK  Ø
13Ø  CIRCLE 45.5,4Ø,14: CIRCLE 17Ø.5,4Ø,14
14Ø  STOP
```

Agora temos um programa com 14 linhas somente, tão fácil de compreender como o anterior mas que ocupa menos espaço na memória do computador, porque tem menos números de linha. Experimente o leitor fazer alguns desenhos. Que tal uma borboleta? Não? Talvez uma cena no espaço sideral? Ou um tanque de guerra? Lembre-se de que pode obter melhores resultados se ensaiar previamente o desenho numa folha de papel, de preferência papel milimétrico. E será muito vantajoso estudar as medidas e diagrama na página 1Ø2 do livro *ZX Spectrum BASIC Programming*.

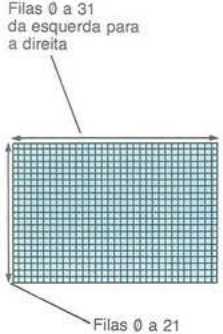
Revisão sumária

Ensinámos a desenhar no computador segmentos de recta entre dois pontos, semicircunferências e circunferências, usando os comandos: PLOT, DRAW, PI e CIRCLE.
Também ensinámos a juntar várias instruções numa só, com o mesmo número, usando o símbolo: (dois pontos).

▼*Experimente o leitor fazer alguns desenhos seus. Recorde que, além do uso de margens e traços a cores, se pode usar a instrução PAPER e a instrução CLS, para colorir o plano de fundo.*



6 Desenhar também com blocos de cores



▲ *Para imprimir blocos coloridos no* écran, *pensa-se nele como se estivesse dividido em filas de quadrados. Há 32 quadrados da esquerda para a direita e 22 quadrados de cima para baixo.*

O leitor aprendeu a desenhar com traços. No entanto, muitas vezes pretendemos pintar toda uma zona. Para o facilitar, a zona do fundo no *écran* está dividida em 22 filas numeradas, da parte superior para a inferior, de 0 a 21. Cada fila tem 32 quadrados numerados, da esquerda para a direita, de 0 a 31. Neste caso, os números mais baixos estão na *parte superior* do papel e os mais elevados na *parte inferior* (no caso do desenho de traços, os números mais baixos são na parte inferior do papel). Pode-se escolher a cor da tinta (o que for impresso) e também a cor do fundo (o plano de fundo) de cada um destes quadrados. Mais importante ainda, cada um destes quadrados pode ser substituído integralmente por qualquer dos símbolos *gráficos* nas teclas numéricas.

Pintar um único quadrado

Escreve-se no teclado:

```
PRINT AT 15,5; INK 3; PAPER 6;"■"
```

PRINT AT é a instrução a usar quando se pretende imprimir qualquer coisa num lugar especial do *écran*. Recordemos que AT está a vermelho na tecla I. Os números 15,5 dizem ao computador que se deseja imprimir no quadrado da fila 15 (a décima sexta a contar de cima), com o número 5 dessa fila (o sexto quadrado a contar da esquerda). Note-se bem a posição dos pontos e vírgulas. E quando chegar ao símbolo gráfico ■, é necessário não esquecer de passar ao modo G. Como já se sabe, passa-se ou sai-se do modo G carregando na tecla CAPS SHIFT com um dedo e premindo uma só vez a tecla 9, com outro dedo. Estando no modo G, se se carregar numa das teclas de 1 a 8 obtém-se um símbolo gráfico. A zona negra do símbolo será pintada com a cor da *tinta* e a zona branca com a cor do *fundo*. Para obter o contrário (zona negra com cor do fundo e zona branca cor de tinta) haverá que carregar em CAPS SHIFT e, só então, premir a tecla com o símbolo gráfico desejado. Não esquecer que para acrescentar as aspas finais é necessário regressar ao modo L. Depois carrega-se em ENTER e vê-se um quadrado próximo da parte inferior esquerda do papel, a metade de cima de cor magenta e a metade de baixo amarela.

Pintar uma fila de quadrados

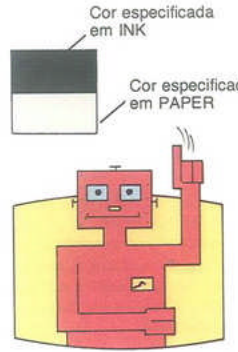
Parece fácil pintar toda uma fila de quadrados, por exemplo, a fila 3 (como se sabe, os quadrados da fila 3 estão numerados de 0 a 31 e da esquerda para a direita). Bastaria escrever:

```
10 PRINT AT 3,0; INK 2; PAPER 5;"■"
20 PRINT AT 3,1; INK 2; PAPER 5;"■"
30 PRINT AT 3,2; INK 2; PAPER 5;"■"
40 PRINT AT 3,3; INK 2; PAPER 5;"■"
```

e assim sucessivamente. O pior é que seria necessário escrever mais 28 instruções semelhantes para colorir toda a fila. Felizmente, há uma forma de fazer isto mais comodamente, usando uma instrução especial para ciclos: FOR... NEXT. O comando FOR está na tecla F e o NEXT na N. Experimente-se teclar estas linhas (TO está a vermelho na tecla F):

```
10 FOR y=0 TO 31
20 PRINT AT 3,y; INK 2; PAPER 5;"■"
30 NEXT y
40 STOP
```

Executando este programa, ver-se-á a fila colorir-se muito rapidamente. O que sucede é isto: As linhas 10 a 30 formam um ciclo FOR... NEXT como a seguir se explica: a primeira vez que o computador chega à linha 10, fica a saber que y tomará valores de 0 a 31. E, porque é a primeira vez, tem que escolher o primeiro valor, que é 0. Por isso, na linha 20, o computador imprime o símbolo gráfico no quadrado 3,0. Depois alcança a linha 30, onde lhe é dito que calcule o valor seguinte (NEXT) de y. Na linha 10, o computador descobre que o valor seguinte é 1. Por isso, na linha 20, imprime o símbolo gráfico no quadrado 3,1. E vai repetindo (iterando) este procedimento até y ser igual a 31. Então o computador não



▲ *Usam-se os símbolos gráficos, que estão no teclado, para colorir os quadrados do* écran. *A área que se vê a negro no símbolo será da cor que for escolhida em INK. E a zona a branco terá a cor escolhida em PAPER.*

▲ *Usando um ciclo FOR ... NEXT, faz-se o computador encher uma fila de quadrados, sem se ser obrigado a dar as coordenadas de cada quadrado.*

encontra mais valores para y, e por isso continua na linha 4Ø onde, finalmente, pára (STOP).

Pintar várias filas de quadrados

Se quisermos pintar, *não só* a fila de quadrados de 3,Ø a 3,31 (todos os quadrados da fila 3), *mas também* as cinco filas por debaixo (de 4,Ø a 4,31; de 5,Ø a 5,31; de 6,Ø a 6,31; de 7,Ø a 7,31; e de 8,Ø a 8,31 — todos os quadrados das filas números 5, 6, 7 e 8), poderíamos fazê-lo com seis grupos de iterações FOR... NEXT com três linhas cada, começando assim (mas não tecle estas linhas):

```
1Ø  FOR y=Ø TO 31
2Ø  PRINT AT 3,y; INK 2; PAPER 5;"■"
3Ø  NEXT y
4Ø  FOR y=Ø TO 31
5Ø  PRINT AT 4,y; INK 2; PAPER 5;"■"
6Ø  NEXT y
```

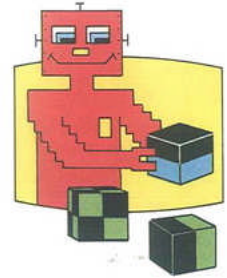
e assim sucessivamente. Por este andar, levaria mais 12 instruções para acabar o bloco de cor. Mas também existe uma maneira muito mais cómoda de o fazer, usando um ciclo FOR... NEXT dentro de outro ciclo FOR... NEXT. Escrevendo no teclado estas linhas:

```
1Ø  FOR x=3 TO 8
2Ø  FOR y=Ø TO 31
3Ø  PRINT AT x,y; INK 2; PAPER 5;"■"
4Ø  NEXT y
5Ø  NEXT x
6Ø  STOP
```

e executando este programa, vê-se o bloco de cor aparecer gradualmente. O que acontece é o seguinte: as linhas de 1Ø a 5Ø formam um ciclo FOR... NEXT. A primeira vez que o computador chega à linha 1Ø, fica a saber que x toma valores de 3 a 8 (números das filas a colorir). Como é a primeira vez que chega a esta linha, é obrigado a escolher o primeiro valor, que é 3. Então agora x é igual a 3. Mas antes de poder continuar este ciclo FOR... NEXT, o computador encontra-se, na linha 2Ø, com ciclo FOR... NEXT que vai das linhas 2Ø a 4Ø. Assim, com x igual a 3, terá que executar todo este FOR... NEXT, isto é, pintará todos os quadrados da fila 3. Quando acabar, passa à linha 5Ø, onde lhe é dito para voltar atrás e calcular o valor seguinte (NEXT) de x. Agora, com x igual a 4, pintará a fila número 4, e assim sucessivamente, até chegar a x igual a 8 para pintar a fila número 8, quadrado 31. Depois não encontra mais valores para x, e por isso segue para a linha 6Ø onde, finalmente, pára (STOP).

Impressão de mais do que um bloco de cores

Se queremos usar vários blocos de cores diferentes num programa, podemos usar o mesmo conjunto de instruções para todos os blocos desde que se ponham como variáveis alguns dos valores dessas instruções. Em seguida, podemos usar esse conjunto de instruções quantas vezes forem necessárias, chamando-o com uma instrução GOSUB, à semelhança do que se fez no programa "SOLDADO" da página 37. Consultando novamente as linhas para imprimir um bloco de cores, vê-se que seis dos valores deste conjunto de instruções podem ser representados por uma variável numérica, e um por uma variável de texto. Então, deixemos (LET):



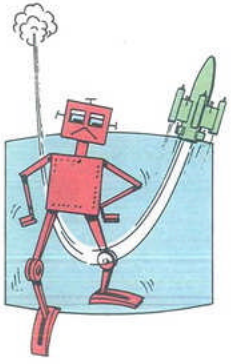
a = ao número da primeira fila com quadrados para colorir.
b = ao número da última fila que tenha quadrados para colorir.
c = ao número do primeiro quadrado pretendido, em cada fila.
d = ao número do último quadrado desejado, em cada fila.
e = ao número a usar depois de INK.
f = ao número a usar depois de PAPER.
g$ = ao símbolo gráfico, entre aspas " ".

Agora pode-se escrever aquele conjunto de instruções assim:

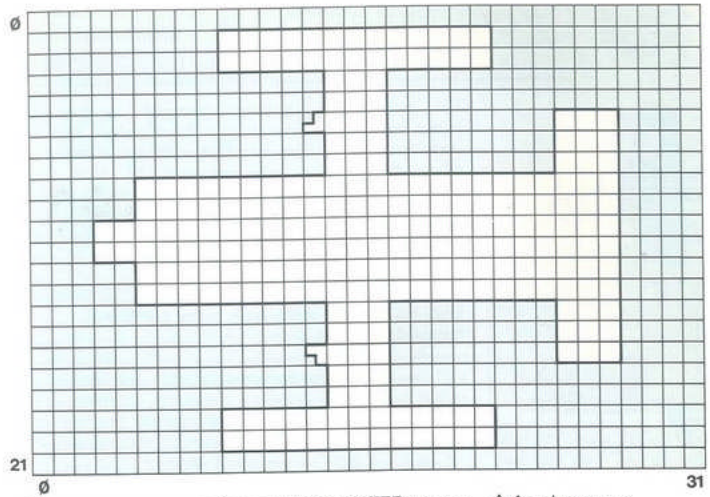
```
1Ø1Ø  FOR x=a TO b
1Ø2Ø  FOR y=c TO d
1Ø3Ø  PRINT AT x,y; INK e; PAPER f;g$
1Ø4Ø  NEXT y
1Ø5Ø  NEXT x
```

Sempre que se queira colorir um bloco num programa, basta definir as variáveis numa linha, e depois outra linha, enviando um GOSUB a este conjunto de instruções. Escreva o programa seguinte no seu teclado. Este programa tem sete blocos de cores e dois quadrados coloridos, isolados (ver página 53 onde estão as instruções para inverter símbolos gráficos):

```
 10  REM "ESPACONAVE"
 20  REM Pintar a fuselagem principal
 30  LET a=8: LET b=13: LET c=5: LET
d=24: LET e=5: LET f=3: LET g$="■"
 40  GOSUB 1000
 50  REM Estabilizadores nas extremidades das
asas
 60  LET a=1: LET b=2: LET c=9: LET d=21
 70  GOSUB 1000
 80  LET a=19: LET b=20
 90  GOSUB 1000
100  REM As asas
110  LET a=3: LET b=7: LET c=14: LET
d=16: LET e=2: LET f=7: LET g$="■"
120  GOSUB 1000
130  LET a=14: LET b=18
140  GOSUB 1000
150  REM Secção da cauda
160  LET a=5: LET b=16: LET c=25: LET
d=27
170  GOSUB 1000
180  REM Sala de comando
190  LET a=10: LET b=11: LET c=3: LET
d=4
200  GOSUB 1000
210  REM Canhoes de laser
220  PRINT AT 5,13; INK 1; "■"
230  PRINT AT 16,13; INK 1; "■"
240  STOP
1000 REM Um bloco colorido
1010 FOR x=a TO b
1020 FOR y=c TO d
1030 PRINT AT x,y; INK e; PAPER f; g$
1040 NEXT y
1050 NEXT x
1060 RETURN
```



Executando o programa, verifica-se se funciona bem. Se não, carrega-se em BREAK SPACE, e depois lista-se o programa no *écran* (LIST e ENTER) para o verificar cuidadosamente.



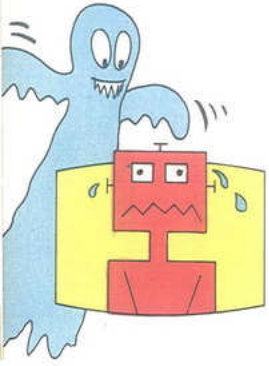
▲*Ao planear os desenhos, é muito mais fácil fazê-los primeiro, assim, numa folha de papel quadriculado.*

Observemos novamente o programa. Na linha 30 foram definidas todas as variáveis. Mas, notará o leitor, na linha 60 só foram definidas algumas delas. Isto é porque as outras teriam o mesmo valor que na linha 30 e, portanto, não é necessário defini-las outra vez. Podemos experimentar agora usar blocos coloridos para desenhar as nossas próprias figuras. Que tal um castelo? Ou um soldado com capacete? Talvez um *robot?* Recorde que, embora seja divertido experimentar desenhar directamente, se obtêm melhores resultados ensaiando, primeiro, o desenho em papel milimétrico. O diagrama na página 102 do livro *ZX Spectrum BASIC Programming* mostra o *écran* em tamanho real.

Revisão sumária

Até agora ensinámos a trabalhar com a instrução PRINT AT, e o uso de ciclos *(loop)* FOR... NEXT. Ou, dito de maneira mais adequada, de um 'sistema iterativo' FOR... NEXT.

7 "Fantasmas"



Este programa é demasiado comprido, porque não passa, basicamente, de um jogo de adivinhas. Há uma casa assombrada com um fantasma numa das suas salas, mas não se sabe em qual. O leitor deve adivinhá-lo! O que torna este jogo emocionante é o facto de o fantasma não estar sempre na mesma sala. Neste programa, o leitor aprenderá também duas coisas novas.

Uso de nomes extensos para variáveis numéricas

Até aqui temos usado letras isoladas em instruções, como em:

```
LET a=2Ø
```

Mas, em lugar disso, podemos usar qualquer palavra, o que muitas vezes ajuda a compreender o que acontece no programa. Por exemplo, pode usar-se:

```
LET A minha altura=5
```

Ou:

```
LET riqueza=2ØØØØ
```

No programa "FANTASMAS", encontra-se na linha 3Ø uma variável chamada *Sala do Fantasma*, e na linha 4Ø a variável *Resultado*. A única regra ao escolher estes nomes é que devem, obrigatoriamente, começar com uma letra.

Utilização de RND

Esta instrução é muito importante e significa que se podem inventar jogos incluindo um elemento de sorte. RND está a verde por cima da tecla T. Também é preciso encontrar o comando INT, que está a verde por cima da tecla R. A instrução seguinte faz o computador imprimir um número inteiro, à sorte (RaNDom), entre 1 e 1Ø:

```
PRINT  1+INT (RND*1Ø)
```

Ou:

```
PRINT  1+INT (RND*8)
```

fará o computador escolher um número inteiro entre 1 e 8, etc. Neste programa, a casa tem oito salas e a sala onde está o fantasma chama-se a *Sala do Fantasma*. Na linha 3Ø, a instrução:

```
LET Sala do Fantasma=1+INT (RND*8)
```

significa que o computador escolhe um número inteiro entre 1 e 8. Até começar um jogo novo, este número será o número da Sala do Fantasma. Mais adiante, na linha 227Ø do programa, o jogador tem de adivinhar esse número. Se adivinhou, o jogo termina em breve, e o computador escolhe outro número, à sorte, entre 1 e 8 para ser a Sala do Fantasma no jogo seguinte.

Se não adivinhou, terá outra oportunidade para tentar a sua sorte. Se adivinhou à primeira, o Resultado será 6. Cada resposta errada diminui o Resultado de 1 ponto. Se baixar a Ø pontos, o fantasma apoderar-se-á de si! Assim, o jogador só tem seis oportunidades para adivinhar o número da Sala do Fantasma.

Agora estudemos o programa cuidadosamente para perceber em que ordem irão suceder os acontecimentos. Usaram-se bastantes GOSUB, que dividiram o programa em partes. Cada uma destas partes deve ser facilmente compreendida. O programa principal termina na linha 13Ø, seguindo-se depois todos os GOSUB.

[Como este programa é muito comprido, talvez se prefira introduzi-lo por partes. Vale a pena recordar que se pode gravar (SAVE) parte do programa e, mais tarde, ler-se (LOAD) outra vez para a memória do computador. No capítulo seguinte, explica-se como se grava (SAVE) e como se lê (LOAD)]. Agora introduz-se o programa e, depois, executa-se:

```
1Ø  REM "FANTASMAS"
2Ø  GOSUB 1ØØØ
3Ø  LET Sala - do - Fantasma=1+INT
```

▲*A instrução INT seguida de (RND*8) fará o computador escolher um número diferente, entre 1 e 8, de cada vez que se execute o programa. O computador escolherá um número, à sorte, de modo que nunca se sabe de antemão qual será esse número.*

```
(RND*8)
  40  LET Resultado=6
  50  GOSUB 2000
  60  IF a <> Sala - do - Fantasma THEN
GOSUB 3000
  70  IF Resultado = 0 THEN GOTO 100
  80  IF a <> Sala - do - Fantasma THEN
GOTO 50
  90  GOSUB 4000
 100  GOSUB 7000
 110  IF b$="s" THEN GOTO 20
 120  IF b$<> "n" THEN GOTO 100
 130  STOP


1000  REM Introducao
1010  BORDER 4
1020  PRINT: PRINT: PRINT
1030  PRINT INK 2; FLASH 1; " - - - - -
- - - - - FANTASMAS - - - - - - - - -
- - - - -"
1040  GOSUB 6500
1050  PRINT: PRINT: PRINT
1090  PRINT: PRINT
1100  PRINT "Como - te - chamas?"
1110  INPUT a$: PRINT a$
1120  GOSUB 6000
1130  PRINT: PRINT
1140  PRINT INK 4;"Em - breve, -"; a$; "-
veras"
1150  PRINT INK 4; FLASH 1; "- - - - -
- UMA - CASA - ASSOMBRADA - - - -
- - -"
1160  PRINT: PRINT: PRINT
1170  GOSUB 6500
1180  PRINT INK 3;"Prime - ENTER - para
- continuar"
1190  INPUT z$: CLS: RETURN

2000  REM Casa do Fantasma e Palpite
2010  REM Casa Principal
2020  LET a=8: LET b=19: LET c=4: LET
d=27: LET e=6: LET f=7: LET g$=■:
GOSUB 5000
2030  REM O telhado
2040  LET a=4: LET b=7: LET c=8: LET
d=23: LET e=2: LET g$="▪": GOSUB 5000
```


```
2050  LET a=6: LET c=6: LET d=7: GOSUB
5000
2060  LET c=24: LET d=25: GOSUB 5000
2070  PRINT AT 7,5; INK 2;"▪":PRINT AT
5,7; INK 2;"▪"
2080  PRINT AT 7,4; INK 2;"▫":PRINT AT
6,5; INK 2;"▫": PRINT AT 5,6; INK 2;"▫":
PRINT AT 4,7; INK 2;"▫"
2090  PRINT AT 7,26; INK 2;"▪": PRINT AT
5,24; INK 2;"▪"
2100  REM Janelas do andar de cima
2110  LET a=9: LET b=12: LET c=5: LET
d=8: LET g$="■": GOSUB 5000
2120  LET c=11: LET d=14: GOSUB 5000
2130  LET c=17: LET d=20: GOSUB 5000
2140  LET c=23: LET d=26: GOSUB 5000
2150  REM Janelas do andar de baixo
2160  LET a=15: LET b=17: LET c=5: LET
d=7: GOSUB 5000
2170  LET c=10: LET d=12: GOSUB 5000
2180  LET c=19: LET d=21: GOSUB 5000
2190  LET c=24: LET d=26: GOSUB 5000
2200  REM A porta
2210  LET b=19: LET c=14: LET d=17: LET
e=3: LET g$="■": GOSUB 5000
2220  PRINT AT 0,0; INK 4; FLASH 1; "- -
- AQUI - ESTA - A - CASA -
ASSOMBRADA - -"
2230  PRINT INK 4; "Ha - 8 - salas. - Qual
- a - do - fantasma,"
2240  PRINT AT 2,8; INK 1;a$;" -?"
2250  GOSUB 6000
2260  PRINT AT 21,0; INK 2; "INTRODUZ -
O - TEU - PALPITE - DE - 1 - A - 8"
2270  INPUT a: CLS: RETURN

3000  REM Resultado depois duma resposta
errada
3010  PRINT
3020  LET Resultado = Resultado -1
3030  PRINT INK 2; "AZAR, -";a$
3040  PRINT:PRINT:PRINT:PRINT
3050  PRINT INK 4; "O - FANTASMA - nao
- esta - na - sala -"; FLASH 1;a
3060  PRINT:PRINT:PRINT:PRINT
3070  IF Resultado = 0 THEN PRINT FLASH
1; BRIGHT 1; INK 2; "ZAS! - O -
```

```
FANTASMA – APOSSOU-SE – DE – TI"
3080  IF Resultado = 0 THEN PRINT:PRINT
3090  IF Resultado = 0 THEN GOSUB 6500
3100  IF Resultado = 0 THEN PRINT INK 1;
"Por – favor – prime – ENTER"
3110  IF Resultado = 0 THEN GOTO 3160
3120  PRINT INK 1; "As – tuas – hipóteses –
baixaram – de – 6 – a –"; INK 2; Resultado
3130  PRINT:PRINT:PRINT:PRINT
3140  GOSUB 6500
3150  PRINT INK 1; "Prime – ENTER – se –
es – bastante –"; INK 2; "bravo"; INK 1;
"para – dares – outro – palpite!!!"
3160  IF Resultado=0 THEN LET a=Sala do
Fantasma
3170  INPUT z$: CLS: RETURN


4000  REM Resultado depois de resposta certa
4010  PRINT PAPER 2; INK 6; FLASH 1;
BRIGHT 1; "– – – – BEM – JOGADO – –
– –"; FLASH 0; " – –";a$: GOSUB 6000
4020  PRINT
4030  PRINT INK 0; BRIGHT 1; " – – – O
– FANTASMA – estava – na – sala –";
FLASH 1; a; FLASH 0; "– –"
4040  REM O FANTASMA
4050  REM Boca do FANTASMA
4060  INK 4
4070  PLOT 103,47
4080  DRAW 52,0: DRAW 0,16: DRAW
-8, -8: DRAW -9,8: DRAW -9, -8: DRAW
-9,8: DRAW -9, -8: DRAW -8,8: DRAW
0, -16
4090  REM Olhos do FANTASMA
4100  INK 0
4110  CIRCLE 110,90,7: CIRCLE 146,90,7
4120  REM Forma do FANTASMA
4130  PLOT 79,25: DRAW 0,60: DRAW 104,0,
-PI: DRAW 0, -60
4140  GOSUB 6500
4150  PRINT:PRINT:PRINT:PRINT:PRINT:
PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:
PRINT:PRINT:PRINT:PRINT:PRINT
4160  PRINT INK 1; "Agora – o – teu –
resultado – foi – "; FLASH 1; BRIGHT 1;
Resultado
```


```
4170  PRINT INK 1; "Obrigado – pelo – jogo,
–";a$
4180  GOSUB 6000
4190  PRINT INK 1; "Agora, – prime –
ENTER."
4200  INPUT z$: CLS: RETURN

5000  REM Um bloco colorido
5010  FOR x=a TO b
5020  FOR y=c TO d
5030  PRINT AT x,y; INK e; PAPER f; g$
5040  NEXT y
5050  NEXT x
5060  RETURN

6000  REM Melodia alegre
6010  BEEP 1,5: BEEP .5,9: BEEP .5,0: BEEP
1,11
6020  BEEP 1,5: BEEP .5,0: BEEP .5,5: BEEP
.5,9: BEEP .5,11: BEEP 1,12
6030  RETURN

6500  REM Melodia sinistra
6510  BEEP 1,5: BEEP .5,9: BEEP .5,0: BEEP
1,11
6520  BEEP 1, -15: BEEP .5, -13: BEEP
.5, -15: BEEP .5, -17: BEEP .5, -15: BEEP
1, -13
6530  RETURN

7000  REM Jogar outra vez ou terminar
7010  BORDER 5
7020  PRINT:PRINT:PRINT:PRINT:PRINT:
PRINT
7030  PRINT INK 3; "Se – quiseres – jogar –
outra – vez – a"
7040  PRINT
7050  PRINT INK 2; "– – – – – – – – –
FANTASMAS"
7060  PRINT
7070  PRINT INK 3;a$; ", – introduz – s –
por – SIM"
7080  PRINT
7090  PRINT INK 3; "– – – – – – – – ou
– n – por – NAO."
7100  INPUT b$: CLS: RETURN
```

▲ *Quando se executa o programa várias vezes, deve-se tentar melhorá-lo.*

Execute o leitor o programa umas poucas vezes e divirta-se tentando caçar o fantasma. Depois pense se pode melhorá-lo:

1) Pode alterar o jogo para que tenha mais interesse? Por exemplo, talvez adicionando linhas que digam ao jogador se o seu palpite foi acima ou abaixo do número correcto, como já se fez num programa anterior, "PALPITES". Poderá introduzi-las entre as linhas 3Ø3Ø e 3Ø5Ø, onde nesta versão há quatro linhas em branco no *écran*.

2) Pode melhorar alguns pormenores do programa? Talvez queira mudar alguns sons, ou a cor da casa.

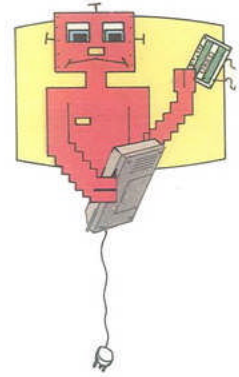
3) Pode melhorar a estrutura genérica do jogo? Por exemplo, talvez queira fazer aparecer o fantasma mais do que uma vez. As linhas 4Ø4Ø e 413Ø, que desenham o fantasma, poderiam tornar-se em algumas linhas GOSUB começando em 8ØØØ. Depois, o leitor poderia chamar as linhas do fantasma entre 3Ø2Ø e 3Ø3Ø, e segui-las com umas poucas notas de música fantasmagórica, de modo a que demorassem no *écran* alguns segundos. Então poderia limpar o *écran*, e seguir para a mensagem de 'Azar, etc.' na linha 3Ø3Ø.

Revisão sumária

Ensinámos neste capítulo a dar nomes extensos a variáveis numéricas. E, o que é mais importante, falámos do uso da instrução 1 + INT[RND*(um número)]. Esta é uma das instruções mais úteis quando se inventam jogos. (Um desafio: será o leitor capaz de mudar o jogo dos 'PALPITES' de forma a jogar *um* só jogador, ou seja, fazendo o computador escolher um número, à sorte, entre 1 e 2Ø?)

8 Gravação e leitura de programas

Gravação



Já aprendemos a dar um título na primeira linha de cada programa, usando um máximo de dez letras ou números, entre aspas. Por exemplo, a primeira linha do programa para cartões de agradecimento foi:

```
1Ø  REM "OBRIGADO"
```

Podem incluir-se espaços em branco no título, desde que a soma total de letras, números e espaços em branco não exceda 1Ø. Para guardar o programa "OBRIGADO" em *cassette*, é necessário:

1) Introduzir a lista das linhas do programa.

2) Escrever no teclado:

```
SAVE "OBRIGADO"
```

O comando SAVE está na tecla S.

3) Desligar, puxando-a para fora, a ficha da tomada EAR, na parte de trás do *ZX Spectrum* (supondo que já antes se tinham ligado entre si as tomadas MIC e EAR do computador e do gravador).

4) Premir ENTER, devendo aparecer imediatamente:

```
Start tape, then press any key.
```

isto significa pôr o gravador em funcionamento e depois premir qualquer tecla.

Se tentou guardar um título com mais de 1Ø números, letras ou espaços em branco, verá outra mensagem:

```
F Invalid file name, Ø:1
```

O nome é incorrecto e terá de recomeçar.

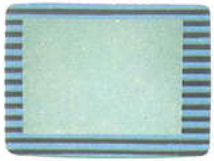
5) Carregar no botão RECORD do gravador (em alguns gravadores também no PLAY).

6) Premir uma tecla qualquer, *excepto* as teclas CAPS SHIFT e SYMBOL SHIFT.

7) Observando agora o *écran*, veremos na zona de enquadramento faixas vermelhas e azuis, depois faixas muito estreitas, azuis e amarelas, depois o *écran* em branco, mais faixas vermelhas e azuis, seguidas das estreitas azuis e amarelas. Depois as faixas devem desaparecer e o computador deve imprimir, cerca do fundo do *écran*, a mensagem:

Ø OK, Ø:1

Verificação da boa gravação do programa



▲*Quando se verifica ou carrega um programa, vêem-se algumas faixas largas azuis e vermelhas no* écran.

▲*As faixas azuis e vermelhas deverão ser seguidas, a intervalos, de aparições de curta duração de faixas estreitas azuis e amarelas.*

1) Volta-se a colocar a ficha na tomada EAR na parte de trás do aparelho.
2) Rebobina-se a *cassette* até exactamente antes do começo da gravação do programa "OBRIGADO".
3) Escreve-se no teclado:

VERIFY "OBRIGADO"

O comando VERIFY está a vermelho, sob o R.
4) Prime-se ENTER, e, de imediato, a zona de enquadramento do *écran* ficará em branco.
5) Carrega-se no botão PLAY do gravador. A zona de enquadramento mudará para entre azul-claro e vermelho, até o computador encontrar o início do programa "OBRIGADO" na *cassette*. Vêem-se então faixas largas vermelhas e azuis, depois faixas muito estreitas azuis e amarelas, seguidas da mensagem:

Programa: OBRIGADO

Depois a margem ficará vermelha, seguindo-se mais faixas muito estreitas azuis e amarelas. Então as faixas terminarão, e o computador deverá imprimir:

Ø OK, Ø:1

O programa estará bem gravado na *cassette*, e podemos desligar o computador sem medo de perder o programa. Se o programa não tiver ficado bem gravado, aparecerá a mensagem R Tape Loading Error, Ø:1. Neste caso deverá repetir-se a operação de gravação.

Leitura de programas

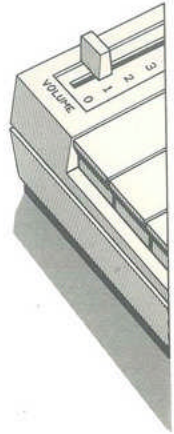
Para ler o programa para o computador:
1) Escreve-se no teclado:

LOAD "OBRIGADO"

O comando LOAD está na tecla J.
2) Prime-se ENTER, e a zona do enquadramento ficará em branco.
3) Rebobina-se a *cassette* até exactamente antes do começo do programa "OBRIGADO".
4) Carrega-se no botão PLAY do gravador. O *écran* parecer-se-á exactamente com o da verificação do programa, e quando aparecer a mensagem de 'OK' no *écran*, é porque o programa foi bem lido e está pronto a ser executado. Mas também poderá aparecer:
R Tape Loading Error, Ø:1,
que significa que o computador não consegue ler o programa. Dever-se-ia então tentar LOAD de novo.

Alguns problemas habituais com a verificação e leitura de programas

1) O volume do gravador demasiado baixo.
2) A cabeça de gravação muito suja.
3) As pilhas do gravador com pouca carga.
4) Início da leitura da *cassette* a meio do programa, em lugar de ter começado mesmo antes do programa.
5) Engano a teclar o nome do programa que se quer verificar ou ler.
6) Esquecimento de ligar a ficha na tomada EAR. Neste caso ensaiar de novo. (Também sucede o esquecimento de desligá-la quando se gravou o programa. Para se ter a certeza, desligam-se os fios do computador à entrada do gravador e *lê-se* a *cassette* na parte da fita onde se pensa que o programa está guardado. Se foi bem guardado, deverão ouvir-se uns ruídos muitos altos!)



▲*Se o volume do som do gravador não estiver regulado devidamente, poderá haver problemas ao carregar programas. Tem de se ajustar o volume por tentativas, antes de encontrar a posição correcta em relação a cada gravador.*

9 Glossário

Aqui se encontram todos os comandos de que já falámos, e também uma breve descrição do seu uso e do que fazem. Este capítulo deve ser lido com cuidado, porque também se encontram nele algumas coisas novas.

BEEP (a vermelho debaixo da tecla Z) Usa-se para criar sons. (Ver também as páginas 135-138 do livro de instruções que acompanha o *ZX Spectrum* intitulado *ZX Spectrum BASIC Programming*.)

BORDER (na tecla B) Quando seguido por uma das teclas numéricas que controlam a cor, mudará a cor da zona ao redor do *écran*. Assim:

```
BORDER 1
```

dará um azul-escuro, etc. (Ver também a página 113 do *ZX Spectrum BASIC Programming*.)

BRIGHT (a vermelho debaixo da tecla B) Existem dois níveis de brilho no *écran*. Depois da instrução:

```
BRIGHT 1
```

num programa, tudo o que estiver impresso no *écran* ficará mais brilhante.

```
BRIGHT 0
```

faz voltar o *écran* ao seu brilho normal. Também se usa BRIGHT dentro de uma instrução PRINT. Por exemplo:

```
PRINT BRIGHT 1; "Feliz";
BRIGHT 0; "- triste."
```

(Ver também as páginas 109-110 do *ZX Spectrum BASIC Programming*.)

CIRCLE (a vermelho debaixo da tecla H) Esta instrução desenhará uma circunferência. É seguida de três números, assim:

```
CIRCLE 50,60,25
```

Os dois primeiros números dão as coordenadas do centro, e o terceiro dá o raio. (Ver também as páginas 123-125 do *ZX Spectrum BASIC Programming*.)

CLS (na tecla V) Esta instrução limpa a zona central do *écran* de qualquer texto ou forma gráfica e cobre-a da cor do fundo que tenha sido previamente especificada. (Se não foi especificada nenhuma cor, essa zona ficará branca.) (Ver também as páginas 25, 37 e 103 de *ZX Spectrum BASIC Programming*.)

DRAW (na tecla W) Esta instrução desenhará um segmento de recta de um ponto anterior num programa para um novo ponto. Por exemplo:

```
DRAW 30,60
```

diz ao computador que desenhe uma linha 30 pontos à direita e 60 para cima da presente posição. Se ainda não foi alcançada nenhuma posição, o computador desenhará a partir do canto inferior esquerdo do *écran*. Esta instrução também pode desenhar uma semicircunferência se usada com PI (a verde por cima da tecla M). Assim:

```
DRAW 30,60,PI
```

desenhará uma semicircunferência no sentido contrário ao dos ponteiros do relógio, do último ponto alcançado num programa a um ponto que está 30 pontos à direita e 60 pontos para cima. (Escrevendo PI com sinal menos (-) faria o computador desenhar a semicircunferência no sentido dos ponteiros do relógio). (Ver também as páginas 121-123 do *ZX Spectrum BASIC Programming*.)

FLASH (a vermelho debaixo da tecla V) Depois da instrução:

```
FLASH 1
```

num programa, tudo o que for impresso a seguir será intermitente, ao mesmo tempo que as cores da tinta e do papel mudam de uma para a outra. A instrução:

```
FLASH 0
```

acaba com os intermitentes. FLASH também pode ser usada dentro de uma instrução PRINT. Por exemplo:

```
PRINT FLASH 1; "SOCORRO!"
```

(Ver também as páginas 109-110 e 122 do *ZX Spectrum BASIC Programming*.)

FOR (na tecla F) Esta é a primeira instrução num ciclo FOR... NEXT. Faz com que o computador execute uma tarefa um certo número de vezes. Assim:

```
10 FOR a = 1 TO 6
20 PRINT a
30 NEXT a
```

imprimirá (em linhas separadas) os números 1, 2, 3, 4, 5 e 6. (Ver também as páginas 31-33 e 41 do *ZX Spectrum BASIC Programming*.)

GOSUB (na tecla H) Os GOSUB são uma forma muito útil de manter os programas bem arrumados e fáceis de seguir. Por exemplo:

```
GOSUB 1000
```

envia o computador à linha 1000 e seguintes, até alcançar a instrução RETURN, que o devolve à linha imediatamente depois de GOSUB donde partiu. Isto significa que uma única linha GOSUB no programa principal pode usar-se para chamar o mesmo grupo de linhas, quantas vezes forem necessárias. (Ver também as páginas 37-38 do *ZX Spectrum BASIC Programming*.)

GOTO (na tecla G) Esta instrução faz o computador saltar para uma dada linha do programa. (Ver também as páginas 16, 25, 31 e 37 do *ZX Spectrum BASIC Programming*.)

IF (na tecla U) Esta instrução usa-se no início de um IF... THEN para dizer que se (IF) alguma coisa for verdadeira, então (THEN) o computador tem que fazer outra certa coisa. Por exemplo:

```
IF a=b THEN PRINT "Certo!"
```

(Ver também as páginas 25-26, 31 e 85 do *ZX Spectrum BASIC Programming*.)

INK (a vermelho debaixo da tecla X) Usa-se com uma tecla numérica para mudar a cor do que está a ser impresso ou desenhado no *écran*, e também a parte da tinta num símbolo gráfico. Assim:

```
INK 3
```

significará que palavras ou traços de desenho, ou ainda a parte da tinta dos símbolos gráficos, aparecerá no *écran* em cor magenta. INK pode usar-se dentro de instruções PRINT. Assim:

```
PRINT INK 3; "Como - estas,-"; INK 1; "Roberto?"
```

(Ver também as páginas 109-111 e 122 do *ZX Spectrum BASIC Programming*.)

INPUT (na tecla I) Esta instrução usa-se para introduzir dados no computador:

1) Pode introduzir números. Por exemplo:

```
50 PRINT "Quanto - se - recebera - para - despesas - diarias?"
60 INPUT b
```

Quando se introduzir uma quantia, por exemplo, 15, e depois se premir ENTER, o número 15 será armazenado numa posição chamada b. E se mais tarde se der a instrução:

```
PRINT b
```

então o número 15 aparecerá no *écran*, mal se carregue em ENTER.

2) Também pode introduzir palavras:

```
210 PRINT "Como - assina - o - seu - nome?"
220 INPUT f$
```

Quando se introduz o nome, por exemplo, Joana, então a palavra Joana será armazenada numa posição chamada f$,

e se mais tarde se der a instrução:

PRINT f$

a palavra Joana aparecerá no *écran*, mal se carregue em ENTER. (Ver também as páginas 16, 25 e 31 do *ZX Spectrum BASIC Programming*.)

INT (a verde por cima da tecla R) Ver mais abaixo RND.

LET (na tecla L) Esta instrução usa-se para armazenar dados no computador:

1) Pode guardar números. Por exemplo, na página 19, usou-se:

LET a=2000

Quando se executou esta instrução, o número 2000 foi armazenado numa posição chamada a. Se se quiser adicionar 6 àquela posição, pode-se usar:

LET a=2006

ou pode agregar outra linha ao programa, que especifique:

LET a=a+6

Também se pode usar uma instrução tal como:

LET Aniversario=1969

2) Pode guardar palavras. Por exemplo, talvez haja uma linha assim:

LET a$="relogio de bolso"

Esta instrução introduz a palavra 'relogio de bolso' numa posição chamada a$. (Ver também as páginas 13, 31 e 38 do *ZX Spectrum BASIC Programming*.)

LIST (na tecla K) Uma instrução útil para fazer o computador listar, no *écran*, o programa que estiver na memória. Se o programa for demasiado comprido para caber todo no *écran*, aparece a mensagem:

scroll?

Para continuar, carrega-se em qualquer tecla, menos em CAPS SHIFT e SYMBOL SHIFT, que nada farão. As teclas N e BREAK SPACE, quando carregadas, param a listagem, de forma a que se possam agregar linhas ou editar as linhas que se vêem no *écran*. Se se quiser listar um programa de certa linha, deverá carregar-se em LIST, seguido do número dessa linha. (Ver também as páginas 15-16 do *ZX Spectrum BASIC Programming*.)

LOAD (na tecla J) Esta instrução usa-se quando se carrega um programa no computador. (Ver também as páginas 141-144 do *ZX Spectrum BASIC Programming*.)

MODOS Já foram completamente explicados no capítulo 1. (Ver também as páginas 7-8 e 193 do *ZX Spectrum BASIC Programming*.)

NEW (na tecla A) Quando se tecla NEW e ao mesmo tempo ENTER, apagam-se todas as linhas do programa que está na memória do computador. Também desaparece tudo o que está no *écran*. (Ver também as páginas 16 e 25 do *ZX Spectrum BASIC Programming*.)

NEXT (na tecla N) Ver, acima, FOR.

PAPER (a vermelho debaixo da tecla C) Esta instrução, seguida por uma das teclas numéricas que controlam a cor, mudará a cor do plano de fundo, por detrás do que se imprime no *écran*. Também mudará a cor da zona do fundo dos símbolos gráficos. Assim:

PAPER 5

significará que o plano de fundo das palavras e símbolos e a zona do fundo dos símbolos gráficos serão coloridos de celeste (azul-claro). Para colorir toda a zona central do *écran*, ver CLS, mais acima. (Ver também as páginas 109-115 e 122 do *ZX Spectrum BASIC Programming*.)

PAUSE (na tecla M) A instrução:

PAUSE 50

pára a execução do programa durante, exactamente, um segundo. (PAUSE 100 daria dois segundos, etc.) (Ver também a página 129 do *ZX Spectrum BASIC Programming*.)

PI (a verde e por cima da tecla M) Ver mais acima, DRAW.

PLOT (na tecla Q) Esta instrução coloca um pequeno ponto no *écran*. Por exemplo:

PLOT 20,100

coloca o ponto nas coordenadas dadas, neste caso, 20 unidades à direita e 100 unidades acima. Depois de uma instrução PLOT, pode-se começar a desenhar a partir do pequeno ponto. (Ver também a página 121 do *ZX Spectrum BASIC Programming*.)

PRINT (na tecla P) Esta instrução usa-se, correntemente, para imprimir no *écran* qualquer coisa. Por exemplo:

PRINT " "

faz o computador imprimir no *écran* tudo o que estiver entre as aspas. Para imprimir uma linha sem nada, basta:

PRINT

Para imprimir a variável a,

PRINT a

faz o computador imprimir o número previamente armazenado na posição chamada a.

PRINT a$

imprimirá as letras previamente guardadas na posição chamada a$

PRINT AT

(AT está a vermelho na tecla I) Imprimirá o que se disse ao computador para imprimir, no quadrado do *écran* cujas coordenadas são dadas depois do comando PRINT AT. (Ver também as páginas 13, 25, 31, 37 e 101 do *ZX Spectrum BASIC Programming*.)

REM (na tecla E) O computador ignora tudo o que estiver depois dum comando REM, que é muito útil para fazer lembrar a finalidade do programa e qual a sua lógica. Por exemplo, no programa "FANTASMAS", tem-se:

1000 REM Introducao
2000 REM Casa do Fantasma e Palpite

etc. Esta instrução também serve para dar um título ao programa, de forma a poder carregá-lo (LOAD) e gravá-lo (SAVE). Por exemplo:

10 REM "OBRIGADO"

No entanto não é necessário dar um título a um programa, usando REM, para que ele possa ser gravado sob um dado nome. Pode até ter-se um título em REM e gravar sob outro nome totalmente diferente. (Ver também as páginas 16, 25 e 31 do *ZX Spectrum BASIC Programming*.)

RETURN (na tecla Y) Ver, acima, GOSUB.

RND (a verde e por cima da tecla T) Esta instrução faz o computador escolher um número à sorte.

PRINT RND

imprimirá um número à sorte entre 0 e 1 (mas nunca chega a 1).

PRINT RND*6

imprimirá um número entre 0 e 6 (mas nunca chega a 6).

PRINT 1+INT(RND*6)

imprimirá um número *inteiro*, à sorte, entre 1 e 6. (Ver também as páginas 73-74 do *ZX Spectrum BASIC Programming*.)

RUN (na tecla R) Esta instrução diz ao computador que execute o programa que está na sua memória. (Ver também a página 14 do *ZX Spectrum BASIC Programming*.)

SAVE (na tecla S) Usa-se quando se grava um programa em fita. (Ver também as páginas 141-144 do *ZX Spectrum BASIC Programming*.)

STOP (a vermelho na tecla A) Esta instrução marca o lugar onde termina um programa. (Ver também as páginas 8, 16, 25 e 34 do *ZX Spectrum BASIC Programming*.)

THEN (a vermelho na tecla G) Ver IF, mais acima.

VERIFY (a vermelho debaixo da tecla R) Usa-se para verificar que a gravação de um programa em fita está bem feita. (Ver também as páginas 141-144 do *ZX Spectrum BASIC Programming*.)

Índice analítico

Algumas palavras, tais como PRINT, estão em muitas páginas. Por isso, para que o leitor não perca o seu tempo, este índice só lista os exemplos com mais utilidade.

B
BASIC 5, 9
BEEP 14-15, 35-38, 63, 68
Blocos de cor 52-57
BORDER 13, 39, 41, 50, 68
BREAK SPACE, uso da tecla 6, 14
BRIGHT 42, 68

C
C, modo das letras maiúsculas 6
CAPS LOCK, uso da tecla 6
CAPS SHIFT, uso da tecla 4-5, 6
Ciclos, ver FOR... NEXT
CIRCLE 49, 68
CLS 29-30, 31, 40, 51, 68
Colorir palavras e planos de fundo 39-44
CONTINUE 14
Cursor 4

D
DELETE, uso da tecla 5
Desenhar circunferências 48-49
Desenhar segmentos de recta 45-46, 62-63
Desenhar semicircunferências 47-48, 63
DRAW 10, 45-49, 62-63, 68

E
E, modo alargado 11-12
Edição 21-23, 26
ENTER, uso normal da tecla 7;
uso como entrada de dados (INPUT) 31, 33
Escrever cartas 28-30

F
FLASH 41-42, 60-63, 69
FOR ... NEXT, ciclos 11, 53-56, 63, 69

G
G, modo das formas gráficas 12
GOSUB 37-38, 56, 60-64, 69
GOTO 13, 14, 69
Gráficos, ver DRAW, Desenhar e G modo

I
IF... THEN, instruções 27-28, 31-34, 60, 62, 69
INK 11, 13, 39-42, 52-56, 60-64, 69
INPUT 20, 24, 29, 31, 60-64, 69-70
INT ver RND

J
Jogos:
Palpites 31-34
Fantasmas 58-64

K
K, modo 4-5

L
L, modo 6
LET 13, 19, 56, 58, 70
LIST 10, 11, 70
LOAD 67, 70

M
Matemática 17-18
MODOS 70 e ver C, E, G, K e L (modos)

N
NEW 10, 12, 70
NEXT 70 e ver FOR ... NEXT

P
PAPER 39-41, 51, 52-56, 62-63, 70
PAUSE 13, 70-71
PI 47-48, 63, 71
PLOT 10, 45-46, 62, 71
PRINT 5, 27, 71
PRINT AT 11, 13, 52-56
Programa, cursor do 21-22

R
REM 21, 32, 37, 71
Resolução de problemas, uso do computador para 17-28
RETURN 71
RND 13, 58-59, 60, 71
RUN 9-10, 71

S
SAVE e gravação de programas 65-66, 71
Scroll (desenrolar) 26
Som nos programas 35-38 e ver BEEP
STOP 9, 71
SYMBOL SHIFT, uso da tecla 7

T
THEN 71 e ver IF ... THEN
Títulos para programas 21

V
Variáveis de texto, 28-33
Variáveis numéricas 18-28, 58, 60
VERIFY 66-67, 71