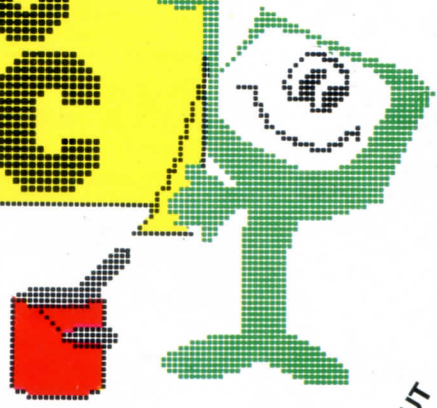
VIDEO BASIC



20 LIÇÕES DE BASIC
PARA APRENDER COM O SPECTRUM

EDIÇÕES LATINAS

JACKSON

- TV e monitor
- Ecran e memória de imagem
- Cores e atributos
- INK, PAPER, BORDER, BRIGHT, FLASH, OVER, INVERSE
- Funções de controlo da impressão em ecran
- TAB, AT
- Números aleatórios com RND e RANDOMIZE
- Utilização avançada de PRINT e INPUT
- Exercícios de vídeo
- Jogo de vídeo n.º 6

6

Spectrum

16K/48K/PLUS

TIMEX COMPUTER 2048

VIDEO BASIC
Uma publicação de:
EDIÇÕES LATINAS-JACKSON
Director editor:
Manuel A. Lopes
Director editor da JACKSON ESPANHA:
Lorenzo Bertagnolio
Director de produção:
Vicente Robles
Autor:
Softidea
Redacção Software:
Francesco Franceschini
Stefano Cremonesi
Desenho gráfico:
Studio Nuovaidea
Ilustrações:
Cinzia Ferrari, Silvano Scolari,
Equipo Galata
Edições Latinas, Lda.
Direcção, redacção e administração, números atrasados e assinaturas:
Av. Almirante Reis, 219, 3.º-Esq.
1000 Lisboa
Fotocomposição e impressão:
Antunes & Amílcar, Lda.

Reservados todos os direitos de reprodução e publicação de desenho, fotografia e textos.
© Grupo Editorial Jackson 1985
© Edições Ingelek 1985
© Edições Latinas 1985
ISBN do tomo 1: 84-85831-12-8
ISBN do fascículo: 84-85831-11-X
ISBN da obra completa: 84-85831-10-1
Depósito Legal N.º: 9963/85
Plano geral da obra:
20 fascículos e 20 cassetes, de publicação quinzenal.
Edições Latinas-Jackson garante a publicação de todos os fascículos e cassetes que compõem esta obra e o fornecimento de qualquer número atrasado, até 1 ano, depois de terminada a sua publicação.
Está reservado ao editor, o direito de modificar o preço de venda dos fascículos durante a sua saída, se o mercado assim o exigir.

2.ª Quinzena – Jan. 1986

EDIÇÕES LATINAS
JACKSON

SUMÁRIO

HARDWARE **2**
Televisores e monitores.
Ecran e memória de imagem.
Os atributos e as cores.

A LINGUAGEM **18**
Funções de controlo da impressão em ecran. AT, RND, RANDOMIZE, TAB.

A PROGRAMAÇÃO **28**
A utilização avançada de PRINT e INPUT.

VÍDEO-EXERCÍCIOS **32**

Introdução

Entre outras coisas, o teu computador é também um emissor de televisão: é capaz de transmitir por cabo as informações que elaboraste. Nesta perspectiva, o programador converte-se no "director" da saída dos dados tratados. Cabe-lhe, portanto, a ele a responsabilidade de facultar às informações de saída o máximo relevo e clareza possíveis. Daí a importância de se conhecer tanto o ***hardware*** *relativo à visualização das imagens como as sentenças e funções do BASIC capazes de manipular o formato e os atributos. O êxito dos teus futuros programas depende da tua familiarização com estes elementos.*

Televisores e monitores

Televisor e monitor ou, mais usualmente, as unidades de visualização constituem o principal dispositivo de saída do computador.

De um modo geral é a estas que se atribui a tarefa de visualizar todas as informações, os dados e as mensagens que constituem a base fundamental da relação entre o homem e o computador.

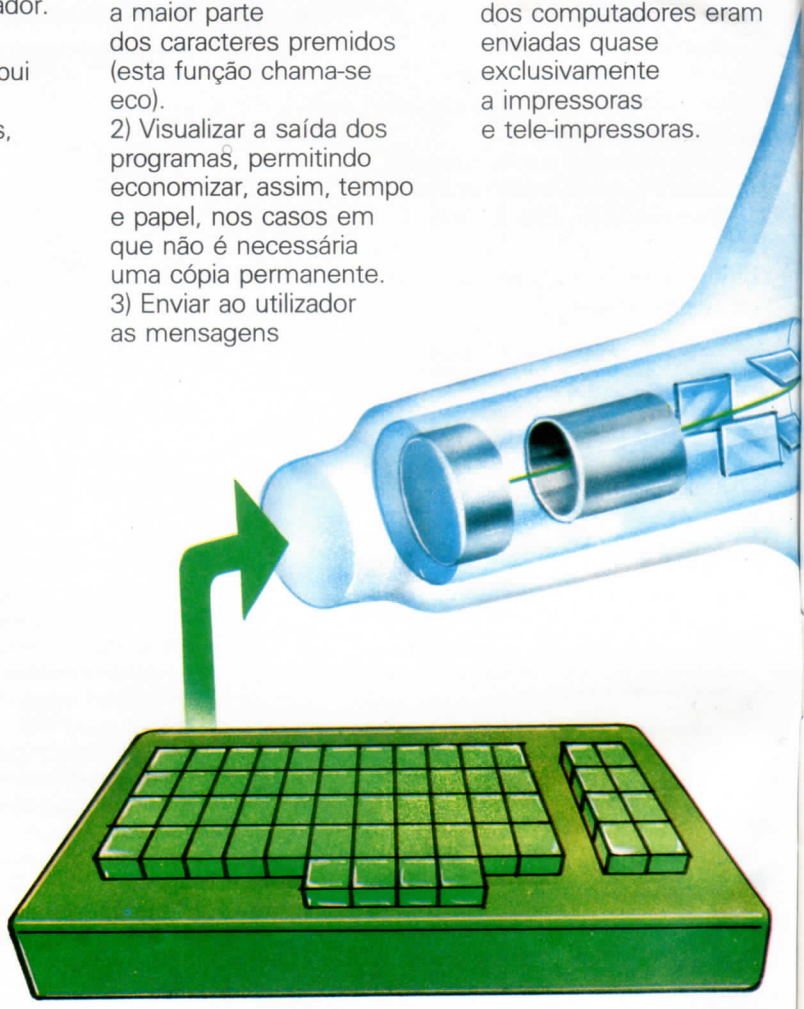
As funções desempenhadas pela unidade de visualização são essencialmente três:

1) Visualizar no ecran a maior parte dos caracteres premidos (esta função chama-se eco).

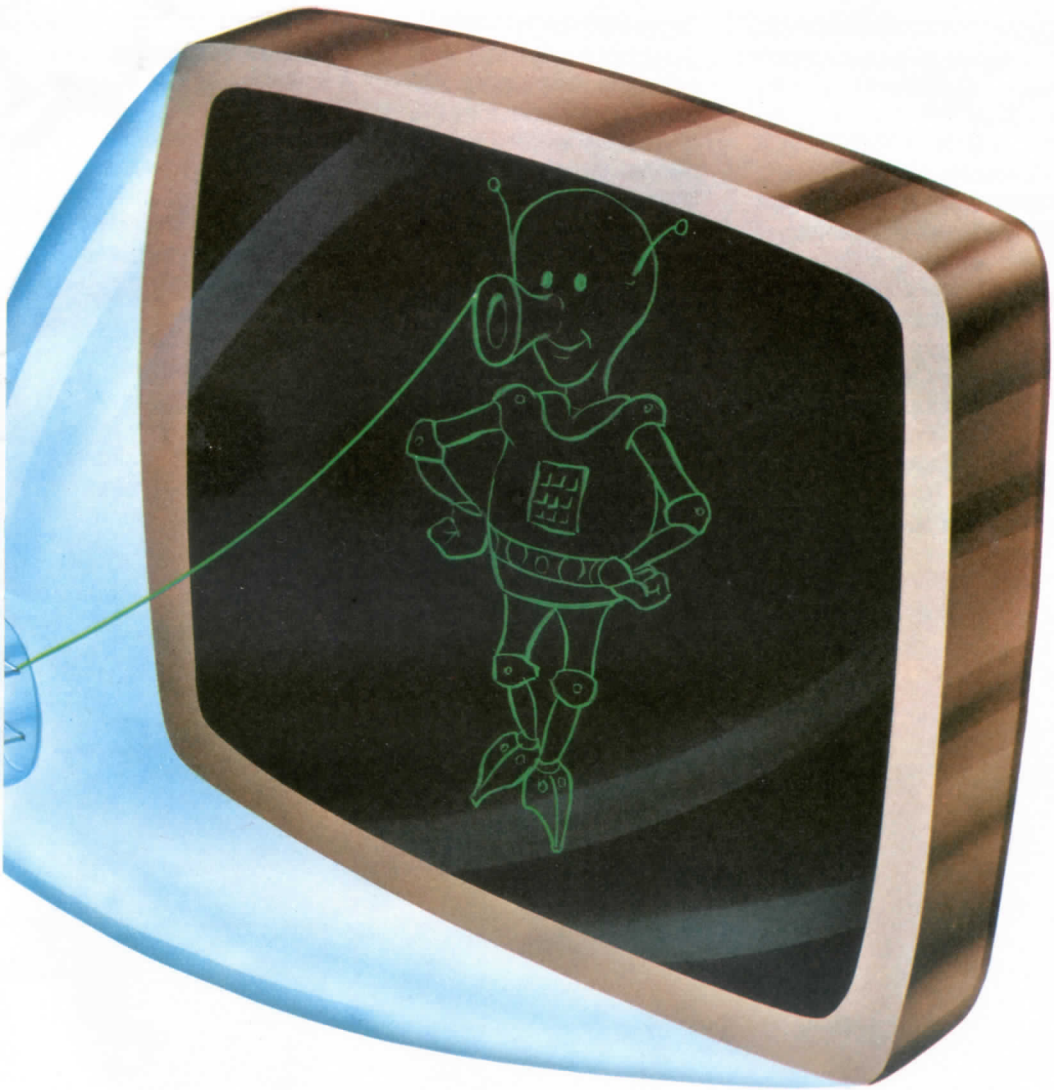
2) Visualizar a saída dos programas, permitindo economizar, assim, tempo e papel, nos casos em que não é necessária uma cópia permanente.

3) Enviar ao utilizador as mensagens do intérprete BASIC (por exemplo, as mensagens de erro).

O emprego de unidades de vídeo como dispositivos de visualização é bastante recente; até há poucos anos ainda, as informações de saída dos computadores eram enviadas quase exclusivamente a impressoras e tele-impressoras.



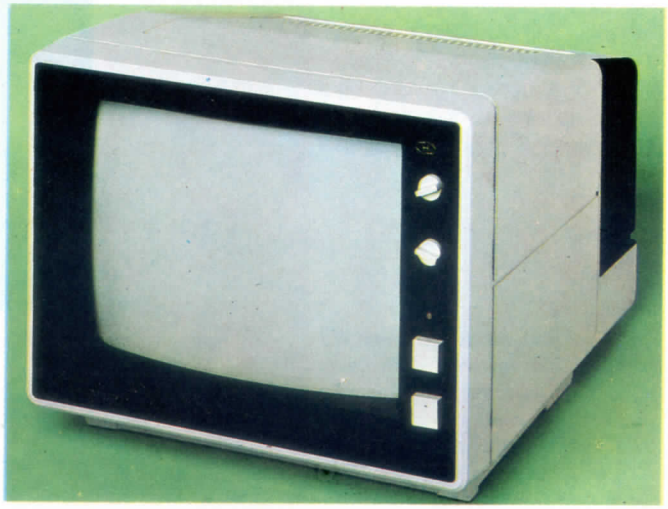
HARDWARE



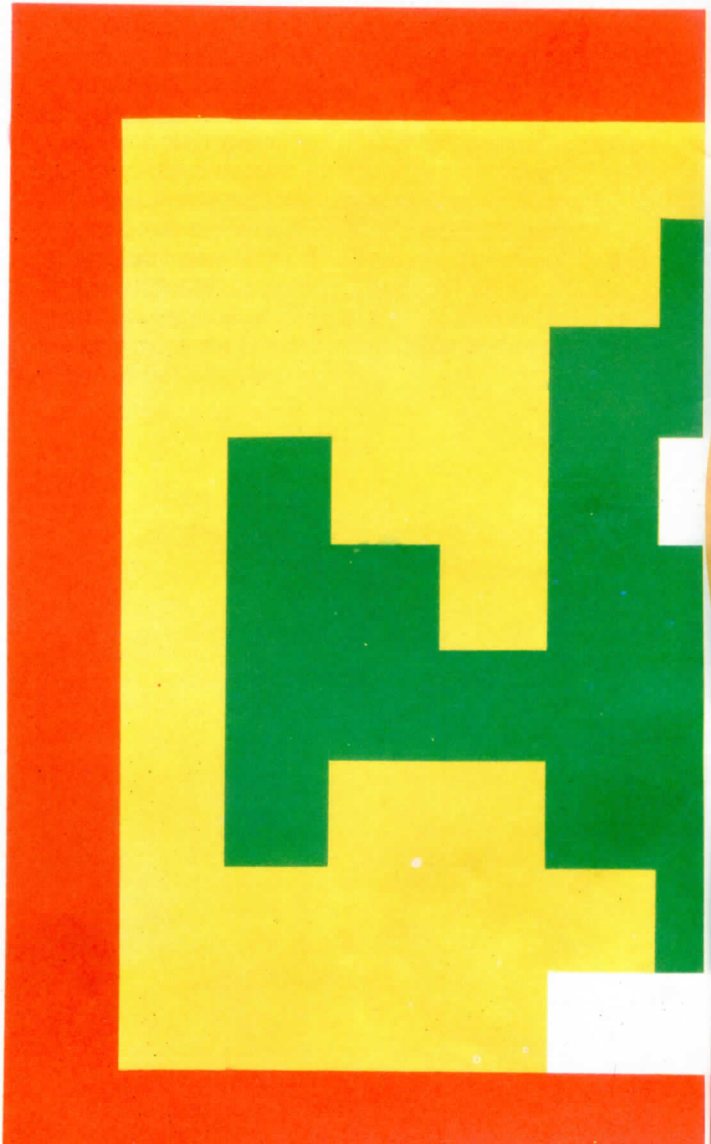
Porém, cedo se verificou que estes dispositivos eram absolutamente insuficientes para fazer frente a um volume crescente de trabalho. Além disso, tinham custos de manutenção muito elevados (eram necessárias montanhas de papel e cuidados permanentes) e possuíam uma capacidade e velocidade de visualização bastante limitadas relativamente às exigências dos utilizadores. E assim pensou-se em complementar as sempre necessárias impressoras com unidades de saída mais adequadas e flexíveis na sua utilização do que as até então usadas. A escolha, como decerto já adivinhaste, recaiu sobre os ecrans de vídeo. Estes correspondiam a todos os requisitos delineados: eram compactos, de confiança, económicos (manutenção escassa e gastos de utilização mínimos), e rápidos. Desde então, o seu emprego foi-se tornando progressivamente cada vez mais intenso; actualmente é quase impossível conseguir encontrar um computador que não esteja dotado com a correspondente unidade de visualização.

HARDWARE

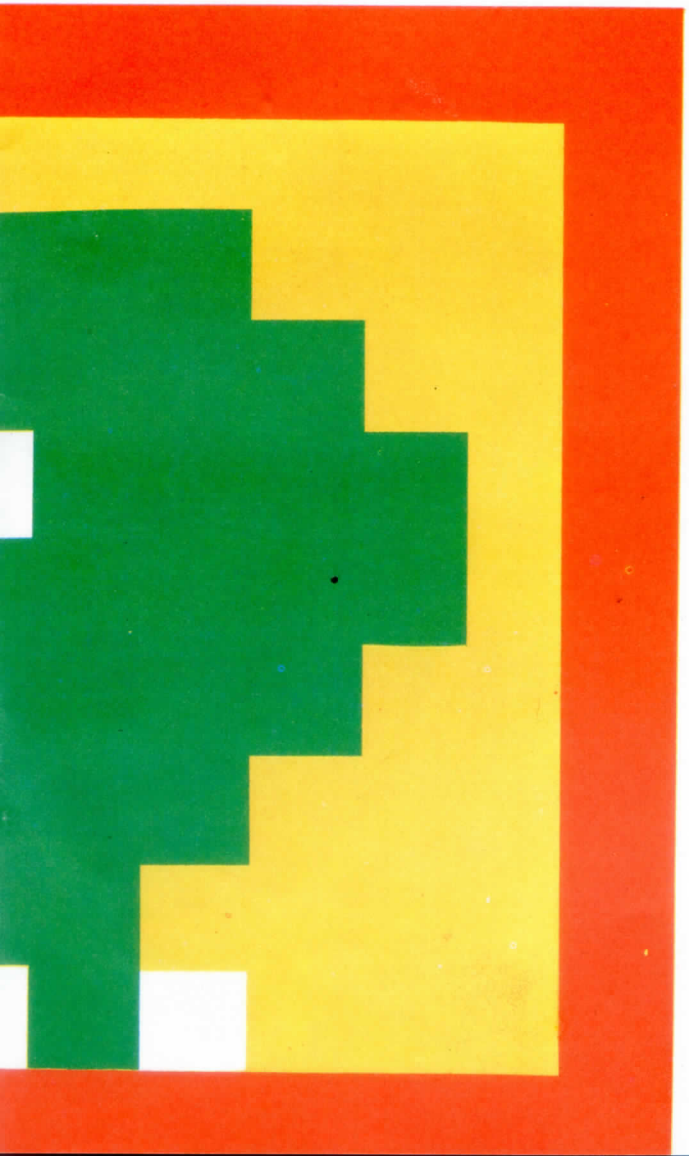
Nos computadores domésticos modernos, as unidades de vídeo habitualmente empregues são de dois tipos: televisores e monitores. Entre um televisor e um monitor não existe, nem física nem substancialmente, uma grande diferença. Um monitor não é mais do que uma televisão de excelente qualidade de que se eliminaram todos os circuítos de recepção do sinal através de antena. Naturalmente, a qualidade da imagem é superior à que se pode obter com um televisor vulgar. Para utilizações não profissionais pode não ser necessário (e até inútil) recorrer à compra de um monitor, visto o televisor doméstico ser perfeitamente capaz de desempenhar o trabalho de visualização a um preço certamente inferior. A forma como se produzem os textos no ecrã do teu *Spectrum* é bastante simples; existe uma zona da memória RAM na qual estão depositados – sob a forma de código ASCII – todos os caracteres que podem ser apresentados em ecran. Um circuito especial, no interior do teu *Spectrum*, faz referência a ela, realizando a sua *interface* com a unidade de vídeo. Este



circuito recebe todos os caracteres presentes nas localizações da memória de ecran e envia-os para o circuito do televisor, sob a forma de impulsos eléctricos compatíveis com o sistema ou *standard* de televisão. O televisor (ou o monitor) produz uma imagem visível a partir destes sinais eléctricos. A maneira como isto se passa constitui uma das aplicações mais interessantes da física electrónica e tem como componente fundamental um dispositivo chamado tubo de raios catódicos, que é um tubo de vácuo, isto é, um recipiente de vidro em cujo interior foi feito o vazio. Dentro deste tubo encontra-se



um "canhão electrónico" (baseado num filamento aquecido pela corrente que o atravessa, como nas lâmpadas) que produz um feixe (ou pincel) de electrões muito delgado. Os electrões gozam de uma propriedade singular: quando chocam com determinadas substâncias fluorescentes – fósforo – levam-nas a gerar uma luminescência, cuja duração pode oscilar entre alguns milissegundos (milésimos de segundo) e alguns segundos, em função do tipo de fósforo e da intensidade do feixe electrónico. A imagem reconstitui-se precisamente devido a este facto: o feixe de electrões "disparado" pelo "canhão" e orientado pelos oportunos campos eléctricos e magnéticos, deslocando-se da direita para a esquerda e de cima para baixo, imprime uma intensidade maior ou menor a cada um dos pontos de um ecran revestido por uma delgadíssima camada de fósforo, provocando uma maior ou menor luminescência. A imagem



A baixa resolução emprega unicamente os caracteres (normais ou gráficos) existentes no teclado. As imagens assim obtidas não são muito pormenorizadas.

é, portanto, reconstituída ponto por ponto pelo pincel electrónico do tubo de raios catódicos, tal como faz o olho humano quando lê a página impressa de um livro ou de um jornal.

A velocidade com que o pincel percorre a totalidde do ecran é de tal modo elevada que não pode ser detectada sequer vagamente pelo observador humano: o *standard* europeu de televisão prevê que todo o ecran, subdividido em 625 linhas horizontais (525 no *standard* americano) seja percorrido completamente em cada cinquenta avos de segundo. O fenómeno da persistência da imagem na retina encarrega-se de proporcionar uma imagem completa e simultânea. No interior

do televisor (ou do monitor) existem, como já referimos, circuitos previstos para guiar os movimentos do feixe de electrões tanto em sentido vertical como horizontal. Para que a imagem seja visível, é necessário que estes dispositivos operem simultaneamente.

No sinal de vídeo estão compreendidas, além das informações relativas à intensidade de cada ponto de ecran, sinais especiais para coordenar o movimento do pincel electrónico. São os sinais de sincronização.

Nas transmissões de televisão, estas informações provêm da câmara; nos computadores, em contrapartida, existe um circuito que, tal como uma câmara, "lê" a imagem que tem de visualizar nas diferentes posições da memória do ecran.

O tipo de fósforo aplicado na superfície do ecran determina a cor do ponto de colisão entre

Em alta resolução podes orientar cada ponto (pixel) determinando ou não a sua luminescência. As imagens assim obtidas são muito pormenorizadas.

o pincel electrónico e o próprio fósforo. Existem no mercado diferentes tipos de monitores: de fósforo verde, amarelo, âmbar, etc. A escolha de uma ou de outra cor depende normalmente de uma questão de gostos e preferências pessoais, embora ultimamente cheguem rumores (e desmentidos) constantes acerca da maior ou menor fadiga visual produzida por um ou outro tipo de fósforo. Em contrapartida, a escolha das dimensões do ecran é muito importante. Um erro em que se incorre com frequência é julgar que quanto maior for o ecran, melhor será a visão.

A imagem, seja qual for a dimensão do ecran, subdivide-se sempre em 625 linhas horizontais: num ecran mais pequeno as linhas serão mais delgadas que num grande. A dimensão óptima do ecran será aquela em que o olho humano, situado a uma distância normal de visão, já não consiga distinguir umas linhas das outras. A dimensão do ecran em polegadas exprime o comprimento da diagonal, indicado precisamente em polegadas: os computadores domésticos e os terminais de vídeo têm normalmente ecrans de 9" ou 12" (9 ou 12 polegadas).

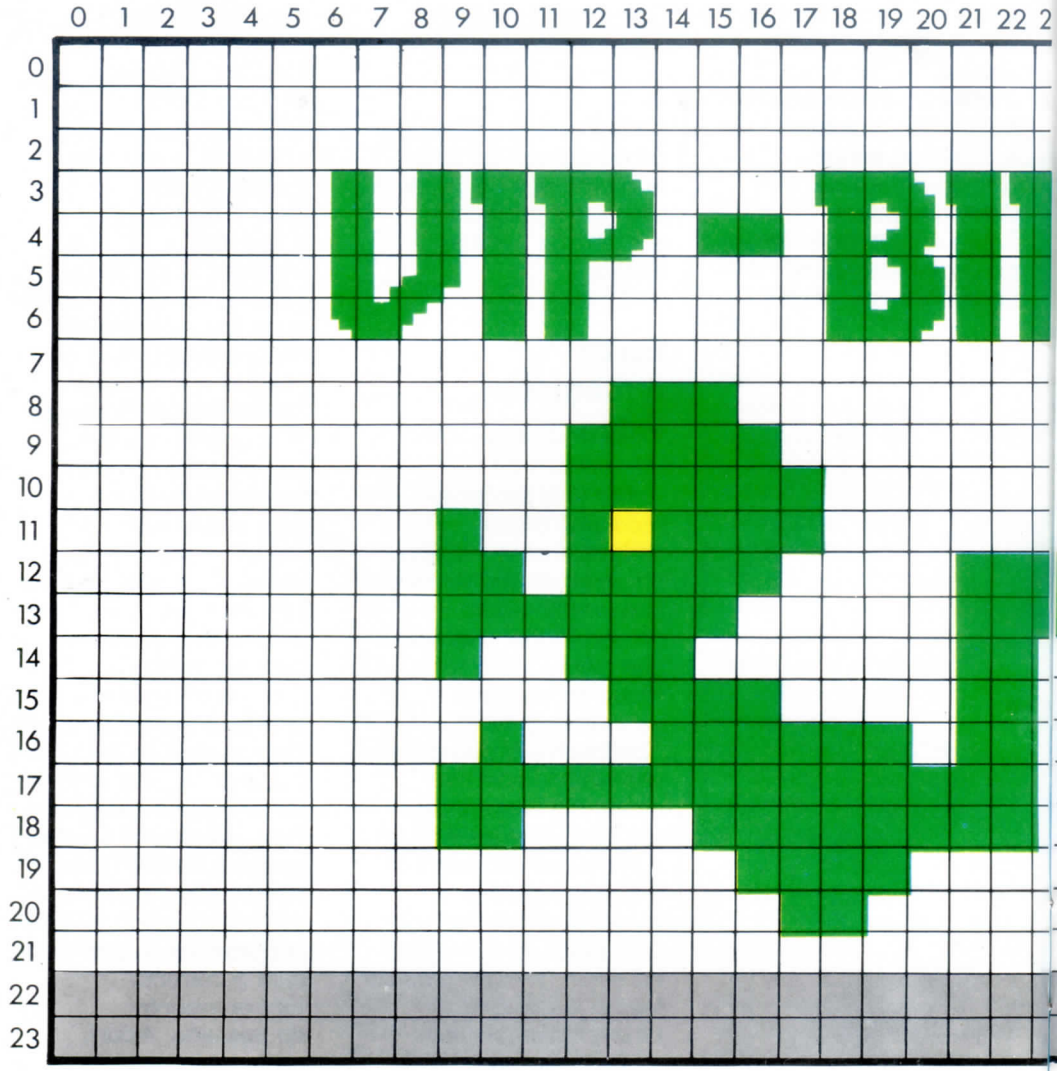
Os ecrans a cores merecem um tratamento à parte.

Permanecendo idênticos os princípios da televisão monocromática, ou seja, de uma só cor (habitualmente chamada "a preto e branco", embora seja verde ou amarela), um ecran a cores baseia o seu funcionamento num fenómeno óptico: todas as cores existentes podem ser obtidas através da mistura e combinação de três cores, por esse motivo chamadas fundamentais: o vermelho, o verde e o azul. No interior do tubo existem três "canhões electrónicos" em vez de um só. A superfície do tubo encontra-se inteiramente revestida por centenas de milhares de pontos de fósforo, dispostos em grupos de três, cada um deles capaz de emitir luz vermelha, verde ou azul. Com o mesmo movimento, já explicado para o ecran monocromático (da direita para a esquerda e de cima para baixo) movem-se agora três feixes electrónicos

em vez de um só; cada um deles – passando através de um ecran metálico adequadamente perfurado – apenas é capaz de atingir o fósforo de uma determinada cor. Sobre a superfície do ecran aparecem portanto três imanges simultâneas, que são respectivamente de cor vermelha, verde e azul e que, combinando-se entre si, proporcionam ao olho humano uma só imagem a cores. Esta técnica é chamada síntese aditiva.

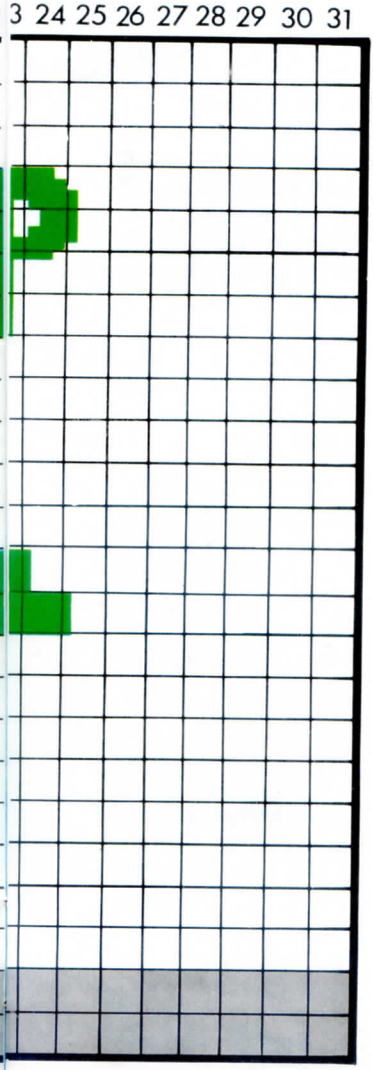
Ecran e memória vídeo

Vimos já o processo mediante o qual as imagens se visualizam no ecran do teu televisor. Trataremos agora de aprofundar a relação que existe entre tudo o que se encontra na memória do teu *Spectrum* e o que podes ver representado no ecran. De facto, existe uma estreita relação entre o conteúdo da memória e a formação das imagens. O computador, para gerar uma imagem no ecran, tem de produzir (como já dissemos) um sinal semelhante ao de uma câmara de televisão, de tal forma que o monitor (ou o televisor) não detecte qualquer diferença. Consequentemente os fabricantes de computadores são obrigados a recorrer a determinados truques para "enganar" o televisor.
Lembras-te de que falámos há algum tempo no mapa de memória do *Spectrum?* Pois bem, chegou o momento de recordar este aspecto. A memória de um computador não está inteiramente diponível para o utilizador: com efeito, existem algumas zonas (ou áreas) que não são directa e livremente utilizáveis para inserir dados e instruções, mas que, no entanto, exercem tarefas que poderíamos chamar de auxílio e apoio. No processo de projecto e constução, foram confiadas a estas parcelas da memória tarefas não exactamente de "elaboração", mas tão importantes quanto estas para o correcto funcionamento do aparelho. Assim, algumas localizações foram destinadas para conter os programas, outras o intérprete BASIC



O ecran do Spectrum está dividido em duas partes: a primeira, de 22 linhas por 32 colunas, disponível para o BASIC, a segunda, de 2 linhas por 32 colunas, reservada para o sistema de operação, os INPUT e as mensagens.

e inclusivamente outras – e estas são as que agora nos interessam – os caracteres de visualização no ecran.



A área da memória de que agora nos vamos ocupar chama-se memória de ecran. O seu objectivo é conter todas as informações necesárias para construir uma imagem no ecran de um televisor. Pensas que esta memória será RAM ou ROM? Não te deveria ser demasiado difícil adivinhá-lo. Se tivermos em consideração que o que queremos é modificar o conteúdo do ecran, a memória vídeo tem forçosamente de pertencer à zona de memória RAM. Nela devem ter lugar as modificações sempre que, por exemplo, executares uma sentença PRINT ou INPUT, ou (excepto em ocasiões especiais) no caso de bateres no teclado qualquer coisa que precise de ser visualizada em ecran. Contudo, por si só a memória de ecran não pode fazer muito: é necessário que um circuito especial de vídeo chamado *refresh* (ou seja, de renovação) da imagem leia – exactamente como uma câmara – toda a memória, carácter por carácter.

A partir desse momento, o circuito de vídeo, que já conhece os códigos dos caracteres a visualizar, consulta uma memória especial ROM (chamada igualmente gerador de caracteres) de que recolhe a "descrição" gráfica do próprio carácter. Como resultado final, obtém-se um sinal de vídeo que indica se cada ponto do ecran deve permanecer aceso ou apagado.

Tudo o que aparece no ecran é representado sob a forma de determinadas combinações de pontos luminosos chamados *pixel* (abreviatura da expressão inglesa *picture element*). Cada ponto, que apenas pode estar apagado ou aceso, pode ser descrito com um único bit. Para não entrarmos em demasiados pormenores técnicos, será suficiente saberes que o circuito de renovação "pega" na combinação de bits que contém a descrição de um carácter e coloca-a na divisão correspondente do ecran, acendendo ou apagando os pontos pelo processo que já indicámos. Os *pixels* luminosos de que dispões no teu *Spectrum* são 49152 ao todo. Ao seu conjunto corresponde um

ecran constituído por 24 linhas e 32 colunas que formam na sua totalidade 24x32 = 768 caracteres.

Os atributos e as cores

O ecran é abstracta e fisicamente divisível em 768 (24 linhas de 32 caracteres) posições onde os caracteres podem ser impressos. Cada um deles é representado por um quadrado de 8x8 pontos. Além disso, cada um destes caracteres pode ser representado de diversas maneiras, tais como, por exemplo: fixo, cintilante, ou colorido. A cor, a luminosidade e a cintilação são os chamados atributos de um carácter visualizado numa dada posição do ecran. Assim, ao imprimires qualquer coisa no ecran não estás senão a modificar a combinação de algum dos atributos anteriormente atribuídos àquela posição. Regra geral, ou seja, quando não indicas especificações concretas – por exemplo, quanto à cor –, o único atributo que sofre alterações é o que corresponde ao aceso ou apagado dos *pixels* que formam o carácter.

Todos os demais atributos permanecem inalterados. Contudo, existe também a possibilidade de modificar a nosso belo-prazer, mediante instruções oportunas, todos os atributos. Vejamos em primeiro lugar a cor. Temos aqui a lista de cores disponíveis no teu *Spectrum*, pela mesma ordem em que vêm indicadas nas teclas numéricas:

---

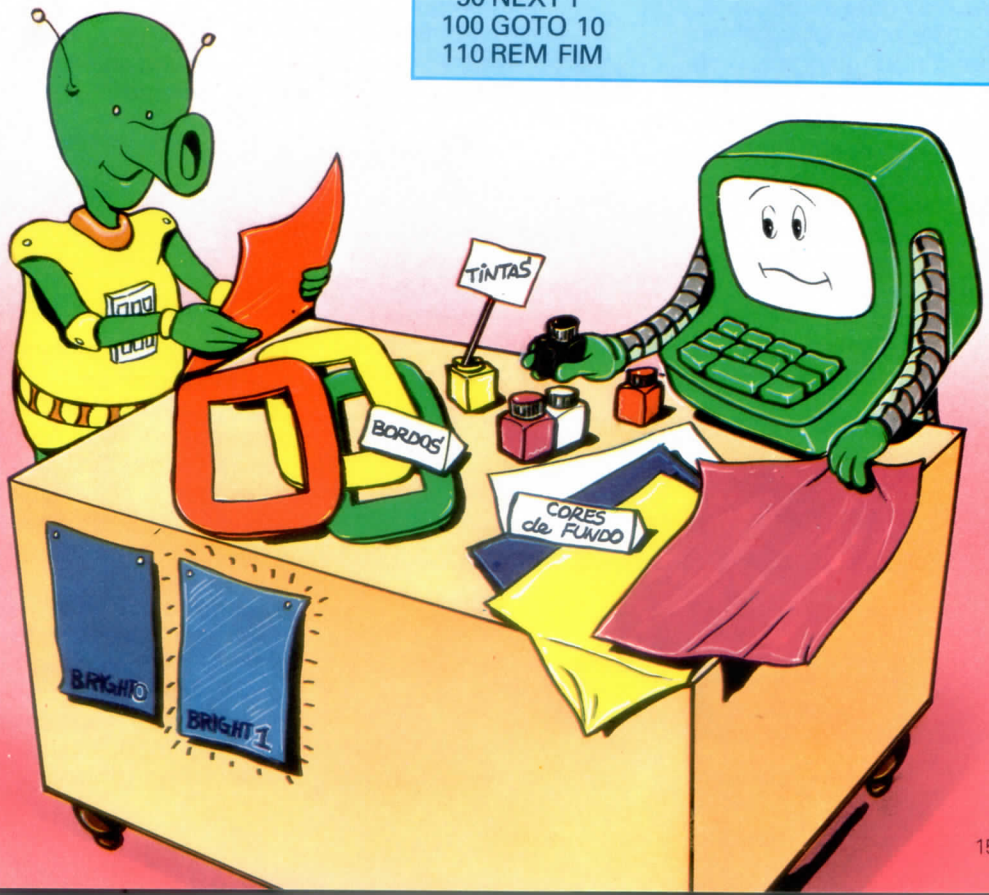
0 – preto
1 – azul
2 – encarnado
3 – púrpura ou magenta
4 – verde
5 – azul turqueza
6 – amarelo
7 – branco

Num televisor a preto e branco os números correspondem a uma escala de cinzentos ordenados do mais escuro ao mais claro. Cada carácter está associado a duas cores: a cor de tintagem, ou cor de impressão, e a cor do papel, ou do fundo. A cor inicial da tinta, no momento do aceso,

é o preto, e a do papel, o branco. Com as instruções PAPER n (papel) e INK n (tinta) é possível alterar as cores.

Para observares o seu efeito, efectua o seguinte programa (para o terminares bastará pressionar um tecla qualquer):

```
 10 FOR I=0 TO 7
 20 PAPER I
 30 FOR J=0 TO 7
 40 INK J
 50 PRINT "PROVA DE IMPRESSÃO"
 60 PAUSE 50
 70 IF INKEY$ <>"" THEN GOTO TO 110
 80 NEXT J
 90 NEXT I
100 GOTO 10
110 REM FIM
```



Portanto, os parêntesis de PAPER e INK (n) podem tomar todos os valores compreendidos entre 0 e 7. No entanto, também é possível atribuir os valores 8 e 9, embora a estes números não corresponda nenhuma cor.
O parâmetro 8 significa "transparente", ou seja, não altera o atributo da posição quando se imprime o carácter, ficando tudo tal como fora anteriormente especificado.
Em contra-partida, o 9 significa "contraste". Consoante a sentença usada, faz-se contrastar a cor da tinta ou do papel, uma com a outra, convertendo-se em branco sobre uma cor escura (preto, azul, encarnado, magenta) e preto sobre uma cor clara (verde, azul turqueza, amarelo ou branco). Além do papel e da tinta, também é possível fazer o bordo do ecran tomar qualquer uma das cores disponíveis. A sentença que efectua esta operação é BORDER n (bordo).
O bordo pode suportar qualquer uma das 8 cores normais (e não 8 e 9).
Este programa simples ensinar-te-á o seu funcionamento:

```
10 FOR I=0 TO 7
20 BORDER I
30 PAUSE 50
40 NEXT I
50 REM FIM
```

As duas últimas sentenças que modificam os atributos são BRIGHT n (luminosidade) e FLASH n (cintilação). BRIGHT n modifica a luminosidade dos caracteres impressos a partir do momento em que se executa a sentença que se segue: n=0 para luminosidade normal, n=1 para luminosidade extra e n=8 para transparência. Se n não for 0, 1 ou 8, ser-nos-á transmitida

a mensagem de que o número indicado não é um valor permitido: no ecran aparecerá escrito INVALID COLOUR (cor não válida).
FLASH n determina se os caracteres impressos a seguir à sentença são intermitentes ou fixos; n = 0 para fixos, n = 1 para intermitentes, n = 8 para a ausência de transformação.
Se n não for um número válido, obteremos igualmente neste caso uma mensagem de erro.
Existem outras sentenças – INVERSE e OVER –, que não modificam os atributos, mas que actuam sobre a matriz de pontos impressa no ecran.
Usam exclusivamente os parâmetros 0 e 1, tal como FLASH e BRIGHT. Como certamente já imaginaste, se empregares a instrução INVERSE 1, o teu *Spectrum* imprimirá a partir desse momento os caracteres em negativo.

Com INVERSE 0 voltarás à normalidade.
Em contrapartida, a sentença OVER 1, activa um tipo especial de sobreimpressão.
Normalmente, ao imprimires um carácter numa qualquer posição do ecran, o símbolo que ali estava anteriormente representado é apagado e substituído por este último. Com OVER 1, pelo contrário, o novo carácter é somado, isto é, sobreposto ao anterior. Isto permite-te, por exemplo, sublinhar um carácter.
Faz a experiência:

```
10 PRINT "A"; CHR$ (8);:REM imprimir o carácter cujo código
   é 8 equivale a voltar atrás com o cursor
20 PRINT "_":REM o símbolo _ obtém-se teclando SYMBOL SHIFT e 0.
```

Em seguida com:

```
10 PRINT "A"; CHR$ (8);
20 PRINT OVER 1; "_"
```

Viste a diferença?

Funções de controlo da impressão em ecran

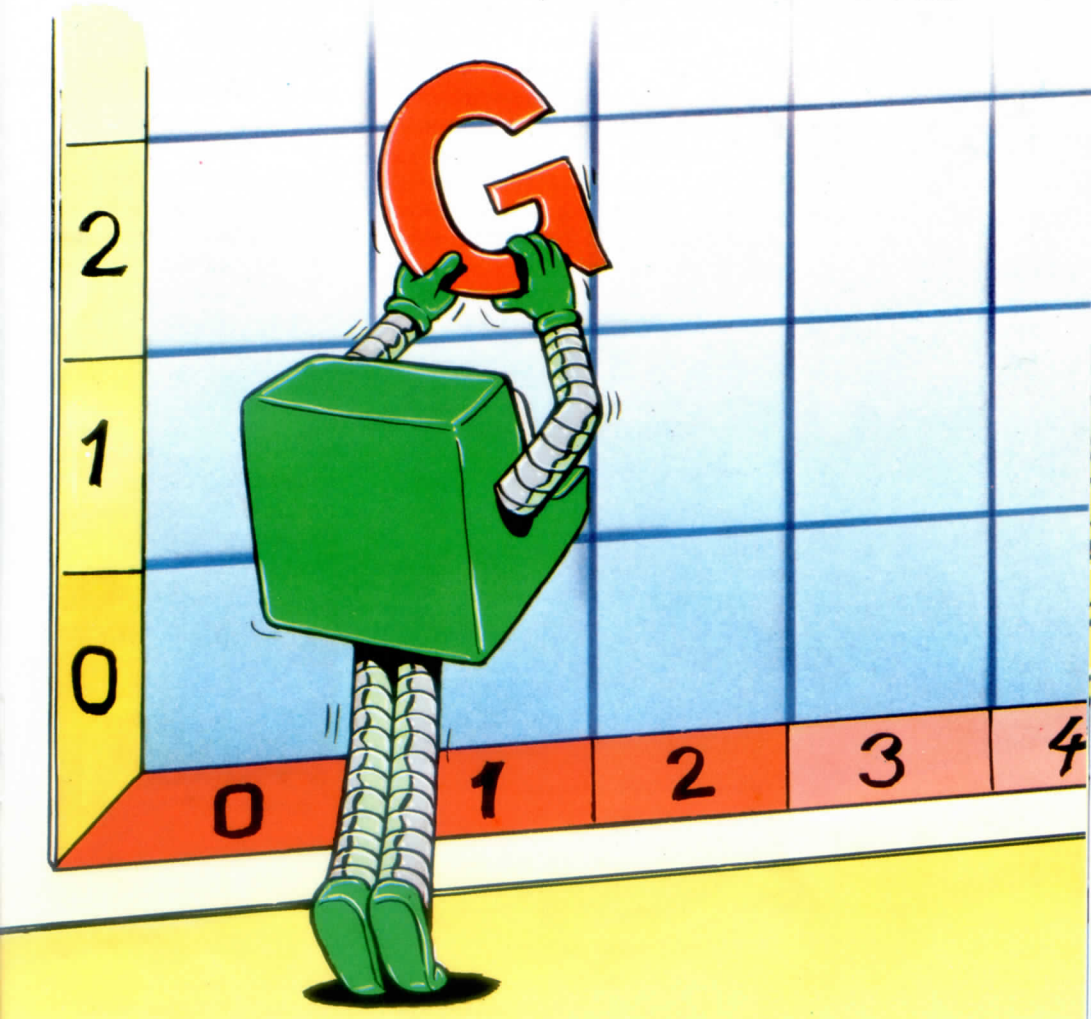
No âmbito das instruções de visualização previstas pelo BASIC revestem-se de uma especial importância todas as sentenças que controlam e modificam, ao gosto do programador, a prosição do cursor e, consequentemente, a dos letreiros em ecran.
Até agora empregámos a instrução PRINT em numerosas ocasiões, utilizando-a para visualizar todos os resultados, as mensagens e os letreiros necessários de cada vez.
Mas o que nos falta ainda é a capacidade de controlar completamente esta instrução, o que nos permitiria, por exemplo, conseguir que os resultados de saída fossem colocados numa determinada posição de ecran, ou então por ordem e em colunas com formatos não impostos necessariamente pelo computador.
Dito de outra maneira e utilizando uma palavra muito usada no campo da informática, desejamos saber formatar os textos em ecran. É, pois, este, o objectivo das instruções que presentemente nos dispomos a aprender.

E é importante acentuar que todas estas instruções não indicam ao computador o que imprimir, mas apenas ONDE imprimir. Compreendeste a diferença?
Analisemo-las agora uma a uma, juntamente com alguns exemplos esclarecedores.

AT

O objectivo da função AT é o de deslocar a posição de impressão (ou seja, o ponto do ecran onde será impresso o carácter seguinte) para a linha e coluna especificadas.

Já vimos que o ecran do teu *Spectrum* destinado ao BASIC está dividido em 22 linhas e 32 colunas. As linhas (de baixo para cima) foram numeradas



de 0 a 21 e as colunas (da esquerda para a direita) de 0 a 31. A linha e a coluna que especificares conjuntamente com a sentença AT deverão ficar compreendidas entre tais intervalos, para não receberes a mensagem de erro INTERGER OUT OF RANGE (fora do ecran). Uma última observação. Já deves ter deduzido que a função AT pode ser extremamente útil em numerosas ocasiões: vê-lo-emos também mais adiante, na parte da lição dedicada à programação. É preciso, porém, ter em conta o facto de com ela se alterarem os mecanismos normais de escrita em ecran, podendo uma sentença AT, indicada sem ponderação suficiente, levar a escrever EM CIMA de um texto já existente em ecran, o que provoca logicamente o seu apagamento. Este facto, que para mais pode ser aproveitado vantajosamente em certas ocasiões, deve ser tomado sempre em consideração para evitar que surjam semelhantes inconvenientes, sempre desagradáveis e aborrecidos.

Exemplos:

`PRINT AT 10, 15; "GATINHO"`	Imprime a palavra "gatinho" exactamente no centro do ecran.
`PRINT AT 21, 27; "GATO"`	Imprime a palavra "gato" no canto inferior direito do ecran.
`PRINT AT 0, 27; "GATO"`	Desta vez o felino aparecerá no canto superior direito.

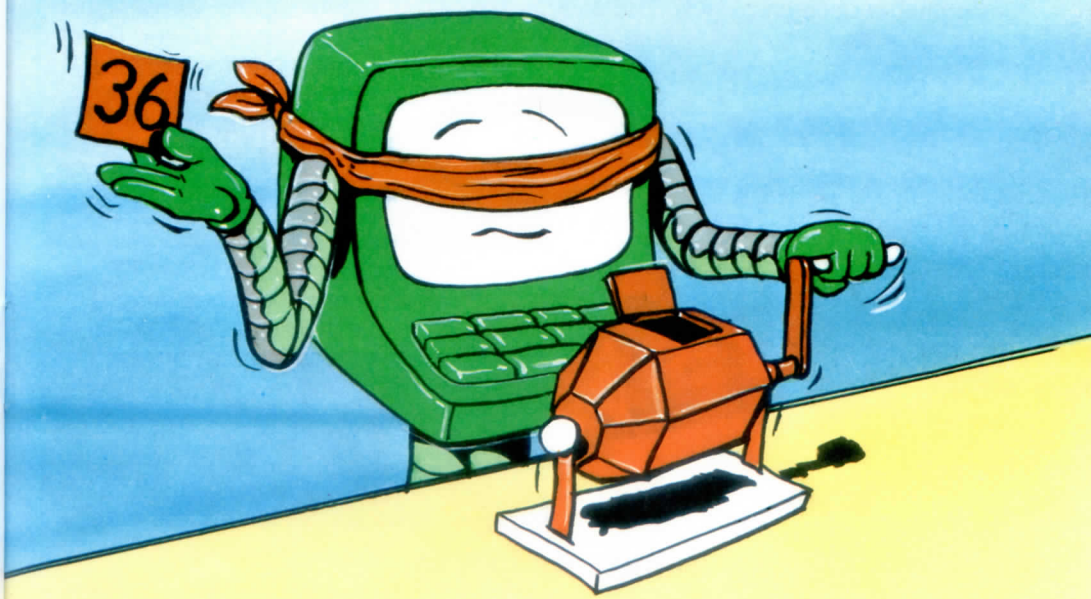
Sintaxe da função

AT linha, coluna

donde a linha tem de ser um número compreendido entre 0 e 21 e coluna entre 0 e 31.

RND

Não é raro depararem-se-nos casos em que se torna necessário dispor de números aleatórios dentro de um programa. Poderás, por exemplo, necessitar de um algoritmo que simule a extracção ao acaso de um número, ou o lançamento de uns dados, ou quem sabe, as cartas saídas num baralho.
A função RND resolve-te perfeitamente este género de problemas: cria números ao acaso (*random* em inglês) compreendidos entre 0 e 1 ($0<=R<1$).
O teu *Spectrum* produz uma sequência de números aleatórios efectuando uma série de operações a partir de um número inicial chamado base, gerado no momento do aceso.
Uma vez que os números gerados por RND são o resultado de uma complexa série de operações, seria mais correcto falar de numeros



pseudoaleatórios. Ao invés das restantes funções, RND não necessita de argumento: portanto PRINT RND é uma instrução correcta e determina a impressão de um número aleatório. Copia e experimenta o seguinte programa para te aperceberes do seu funcionamento:

```
10 FOR C=1 TO 100
20 PRINT C; " "; RND,
30 NEXT C
```

Dado que a gama de números compreendidos entre 0 e 1 não é apropriada para a maioria das aplicações, emprega a seguinte fórmula para estabeleceres por ti mesmo a hierarquia por que devem ser produzidos os números aleatórios:

```
LET R=INT ((LS-LI+1)*RND)+LI
```

em que R significa número aleatório (*random*), LS é o limite superior da hierarquia e LI o limite inferior. Para obter números aleatórios inteiros compreendidos entre 9 e 18, aplicando a fórmula, terás:

```
LET R=INT ((18-9+1)*RND)+9
```

Para números compreendidos entre 0 e 10, será:

```
LET R=INT ((10-0+1)*RND)+0
```

ou seja:

```
LET R=INT (11*RND)
```

Sintaxe da função

RND

RANDOMIZE

A instrução RANDOMIZE usa-se fundamentalmente para determinar qual o ponto da sequência a partir do qual a função RND tem de começar a extrair números aleatórios. Experimenta agora com:

```
10 RANDOMIZE 22
20 FOR C=1 TO 10
30 PRINT RND
40 NEXT C
```

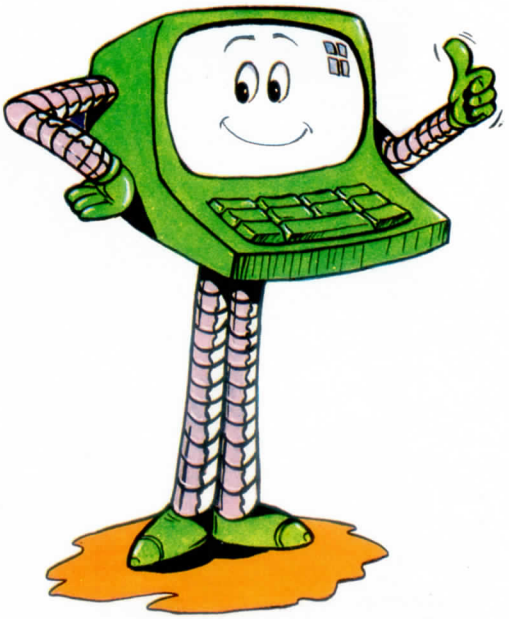
Anota os dez números aleatórios, em seguida põe de novo o programa em andamento com RUN e compara-os. Modifica agora na linha 10 o número que se segue a RANDOMIZE e torna a tentar... Notaste a diferença? Modifica de novo a linha 10. Escreve:

```
10 RANDOMIZE
```

ou então

```
10 RANDOMIZE 0
```

sem alterar as restantes linhas. Torna a pôr o programa em andamento várias vezes, anotando e comparando

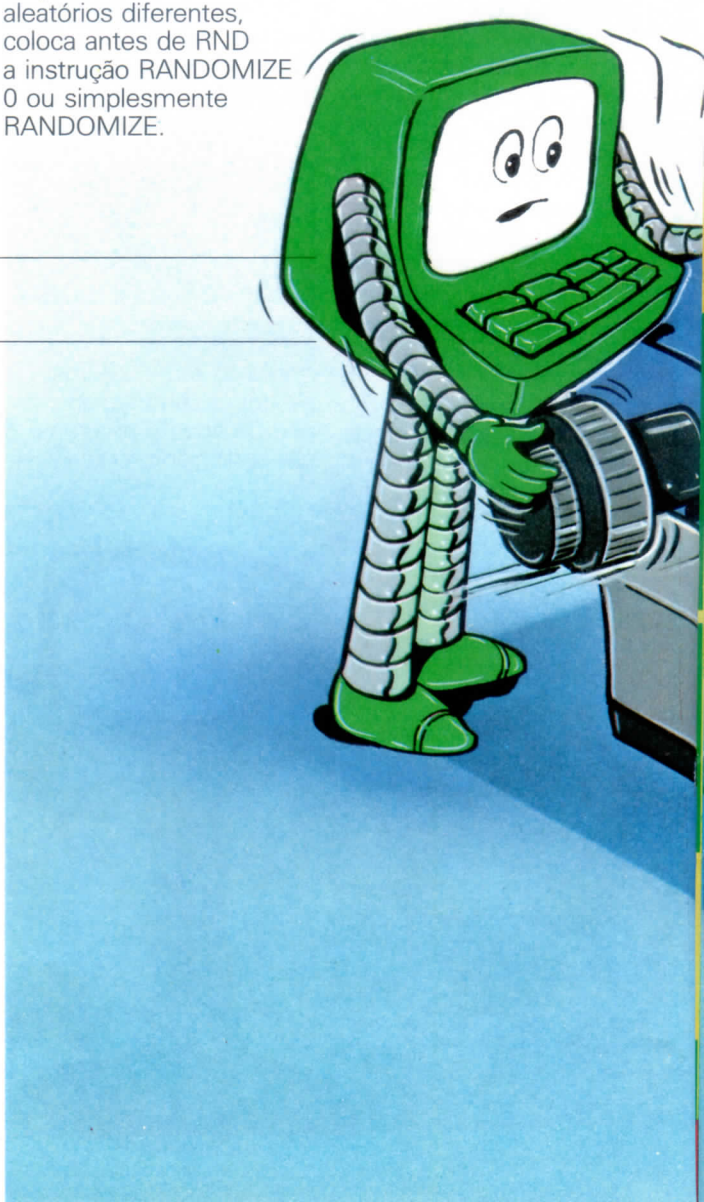


as diferentes séries de números: nunca encontrarás uma idêntica à outra. Resumindo:

- Se pretendes gerar séries de números aleatórios iguais entre si, coloca antes da função RND uma instrução RANDOMIZE, sempre seguida pelo mesmo número.
- Se, pelo contrário, pretendes obter sempre sequências de números aleatórios diferentes, coloca antes de RND a instrução RANDOMIZE 0 ou simplesmente RANDOMIZE.

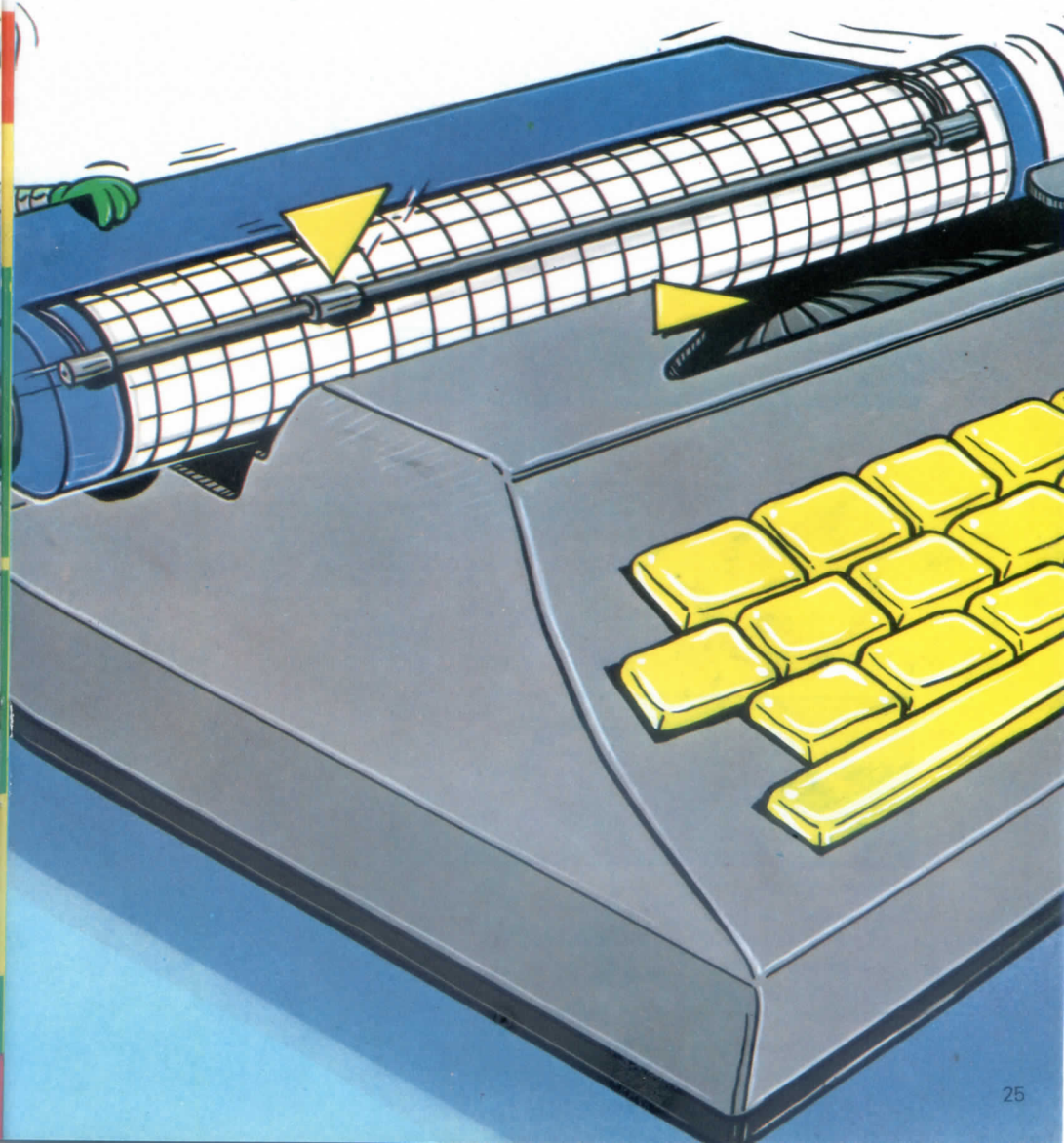
Sintaxe da instrução

RANDOMIZE [número]



TAB

À semelhança de AT, a função TAB () também se emprega apenas no interior de PRINT: opera como os tabuladores de uma simples máquina de escrever.
A função TAB () usa-se normalmente para alinhar as impressões em colunas verticais,



alinhando-as em pontos pré-estabelecidos. Supõe que precisas de visualizar determinados dados em ecran, alinhados numa determinada ordem. Graças a TAB () podes evitar os aborrecimentos e por vezes complicados cálculos para dispôr com exactidão numa coluna todos os elementos. TAB () permance na mesma linha em que se encontra o cursor, excepto quando a posição de impressão especificada não seja anterior à actual posição. Nesse caso deslocar-se-á para a linha seguinte.

Exemplos:

```
PRINT NOME$; TAB (13); APELIDO$
```

Com esta instrução, e independentemente do comprimento da cadeia contida em NOME$, obter-se-á a impressão das variáveis com a seguinte disposição: a primeira a partir da coluna 0, a segunda a partir da 13.

```
10 PRINT TAB (2); "NÚMERO"; TAB
   (12); "QUADRADO"
20 FOR I=1 TO 15
30 PRINT TAB (4); I; TAB (15); I ↑ 2
40 NEXT I
```

Este programa mostra uma aplicação simples de TAB: imprime em colunas os primeiros 15 números com os respectivos quadrados.

Sintaxe da função

TAB (expressão)

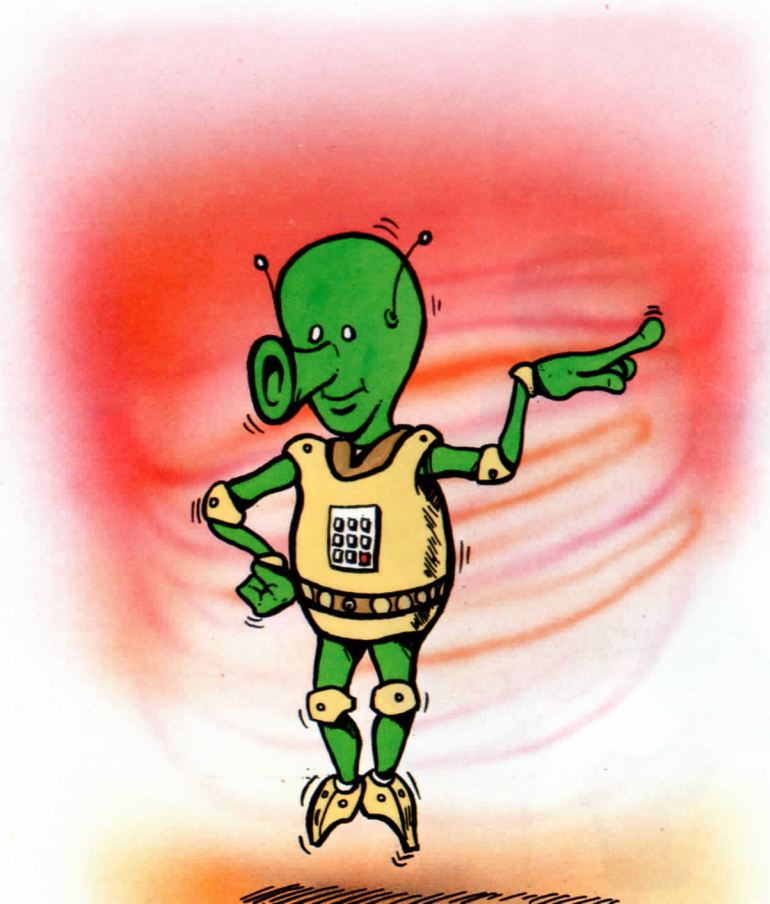
LINGUAGEM

O erro está sempre à espreita: espera maldosamente as tuas distracções, até as mais pequenas. Nunca lhe dês a oportunidade de se rir de ti. Aprende a sintaxe das intruções. Realiza os programas sob a forma de módulos, facilmente legíveis e muito documentados e presta a máxima atenção quando os digitas: até uma vírgula fora de sítio pode comprometer a correcta execução de um programa.

Utilização avançada de PRINT e INPUT

Vais agora aprender a usar vantajosamente as várias instruções de ENTRADA/SAÍDA que fomos vendo. Vamos falar da utilização avançada de PRINT e INPUT.

A palavra "avançada" permite deduzir as suas numerosas possibilidades e, de facto, indica o conjunto de regras, segredos e truques que permitem ao programador obter o controlo pleno



e absoluto de tudo o que entra ou sai do computador. Uma primeira e muito útil instrução será:

```
POKE 23692,255
```

isto evitará que a máquina, depois de ter enchido o ecrã com letras ou números, escreve sempre SCROLL? na linha inferior, parando consequentemente a execução do programa. Tenta, por exemplo:

```
10 FOR X=65 TO 127
20 PRINT X; TAB 8; CHR$ (X)
30 POKE 23692,255
40 NEXT X
50 IF INKEY$ <> "" GO TO 10
```

Descobrirás que a visualização dos caracteres criados no interior do anel FOR pela sentença CHR$ não sofrerá nenhuma interrupção durante a sua execução. A única maneira de interromper o programa (linha 50) será pressionado uma tecla qualquer. Para restabelecer o procedimento normal de SCROLL tecla:

```
POKE 23692,1
```

Vejamos agora algumas coisas acerca de INPUT. Esta instrução é capaz de fazer mais algumas coisas do que as que viste até agora. Supõe que pretendes escrever um programa para aprender a fazer multiplicações de cabeça. Pretendes que como entrada te seja pedido um par qualquer de números e que em seguida apareça uma pergunta a que deverás responder com o resultado da multiplicação. O problema será o seguinte: os números de entrada serão atribuídos a duas variáveis (por exemplo, A e B). Como é que poderemos atribuir a uma terceira variável (chamemos-lhe C) o número proporcionado como resultado, mediante uma sentença INPUT? Uma linha de programa como:

```
INPUT "QUANTO
DARÁ";A;"*";B;"?";C
```

consideraria todos os elementos fora das aspas como variáveis a atribuir. Esta dificuldade

(tão trabalhosa de explicar!) supera-se, em contrapartida, rápida e facilmente: bastará encerrar entre parênteses os nomes das variáveis:

```
INPUT "¿QUANTO DARÁ"; (A);"*"; (B);"?"; C
```

E já esta tudo resolvido! É bem simples e esta regra é geral. Qualquer expressão, que comece por uma letra e que deva ser impressa como parte de uma mensagem, deve ser encerrada entre parêntesis. Assim, o programa completo poderia ser deste estilo:

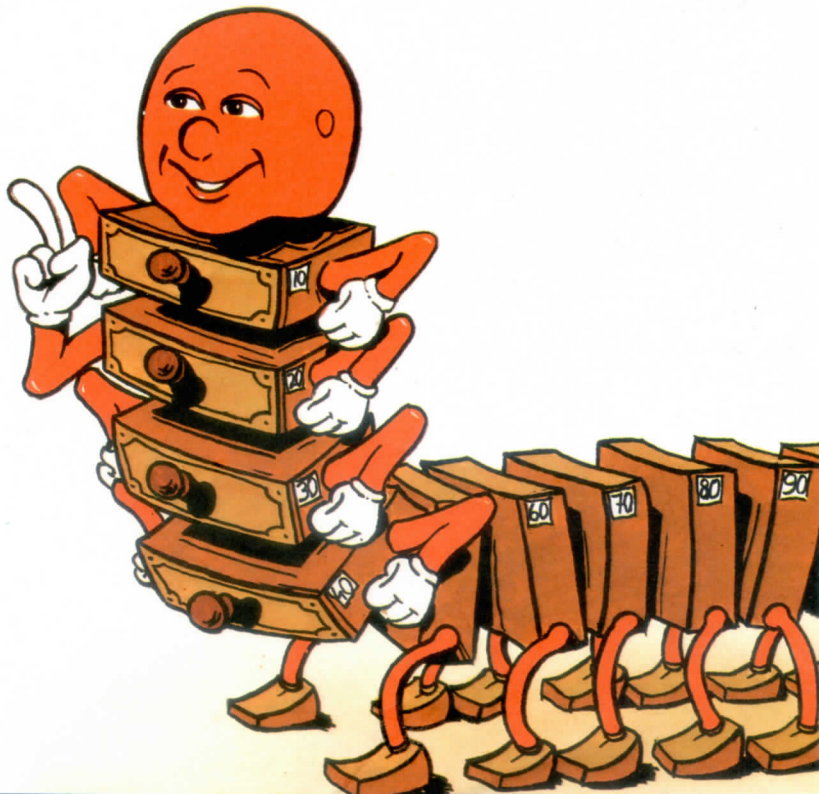
```
10 LET LINHA=10
20 CLS: INPUT AT 10, 0; "¿QUAL É O PRIMEIRO NÚMERO?"; A
30 INPUT AT 12, 0; "¿QUAL É O SEGUNDO NÚMERO?"; B
40 CLS INPUT AT LINHA, 0; "¿QUANTO DÁ"; (A); "*"; (B); "?"; C
50 IF C=A*B THEN PRINT AT LINHA+4, 0; "BEM!!": GO TO 100
60 PRINT "ERRO!!"
70 LET LINHA=LINHA+2
80 IF LINHA<=14 THEN PRINT TAB 15; "TORNA A TENTAR";
   PAUSE 100: GO TO 40
90 PRINT AT 21,0; "O RESULTADO DE"; A; "*"; B; "É", A*
100 INPUT "¿QUERES CONTINUAR?"; R$:IF R$="S" THEN RUN
110 REM FIM
```

A listagem não requer demasiados comentários: a única questão relevante surge na linha 80. Esta linha é constituída por várias instruções (devidamente separadas pelos dois pontos), a primeira das quais é uma IF. Tem em conta que se a condição de IF

for falsa (quer dizer, não verificada), naquela linha não se executará nenhuma das restantes instruções: o intérprete de BASIC salta toda a linha, conduzindo a execução para a instrução seguinte (ou seja, para a linha 90). Outra coisa, desta vez tanto acerca de PRINT como de INPUT: por vezes sucede desejar-se imprimir em ecran uma palavra encerrada entre aspas. No entanto, isso não será possível em condições normais.

As aspas (" ") são usadas em BASIC como delimitadores: sempre que o intérprete BASIC encontra uma considera-a como carácter inicial ou final de uma constante de cadeia. Que fazer então? A resposta chama-se código ASCII. Bastará empregar o código ASCII das aspas (que é o 34) para se "saltar" o intérprete BASIC. Imagina que desejamos escrever em ecran a frase: "O que é bem feito, bem parece". A instrução para a visualizar em ecran entre aspas será:

```
PRINT CHR$ (34); "O QUE É BEM FEITO, BEM PARECE"; CHR$ (34)
```



EXERCÍCIOS

Empregando a função TAB, procura um novo método para imprimir a tabuada de multiplicar de um número, de forma a que as unidades, as dezenas, etc., dos diferentes produtos fiquem perfeitamente alinhadas em colunas. Para te ajudar, eis aqui um exemplo com a tabuada dos 7.

```
10 LET C=10: LET S=C: REM AS UNIDADES IMPRIMEM-SE NA COLUNA 10
20 CLS
30 FOR V=1 TO 10
40 LET P=V*7
50 IFP>9 THEN S=C-1
60 PRINT TAB S; P
70 NEXT V
```

Substitui agora a linha 30 por: 30 FOR V=1 TO 20
Sugestão: para alinhar em colunas as centenas, introduz entre a linha 50 e a 60 um novo controlo.

Joga aos dados com o teu "Spectrum"

```
 10 CLS
 20 INPUT "JOGADOR 1="; J$
 30 INPUT "JOGADOR 2="; H$
 40 PRINT AT 1, 10; "DADO LOUCO"""
 50 PRINT J$; "PRIME UMA TECLA"
 60 PAUSE 0
 70 LET R1=INT (6*RND)+1
 80 PRINT AT 10; R1;"
 90 PRINT H$; "PRIME UMA TECLA"
100 PAUSE 0
110 LET R2=INT (6*RND)+1
120 PRINT TAB 10; R2
130 IF R1=R2 THEN PRINT"; TAB 4; "EMPATE": GO TO 200
140 IF R1>R2 THEN PRINT"; TAB 4 "GANHA"; J$: GO TO 200
150 PRINT"; TAB 4; "GANHA"; H$
200 PRINT"; "CONTINUAMOS? S/N"
210 PAUSE 0: LET T$=INKEY$
220 IF T$="S" OR T$="s" THEN CLS: GO TO 40
230 CLS: PRINT AT 12, 14; "FIM"
```

Modifica as linhas 70 e 110 para simular o lançamento de dois dados: pensa quais são os números máximo e mínimo que podem sair.

BOLETIM DE ASSINATURA

Envie a **EDIÇÕES LATINAS – VIDEOBASIC**
Av. Almirante Reis, 219, 3.º-Esq. • 1000 LISBOA

Desejo subscrever o curso completo de VIDEOBASIC por 7.500$00.

Junto envio cheque n.º_____ Banco_____

Nome

Morada

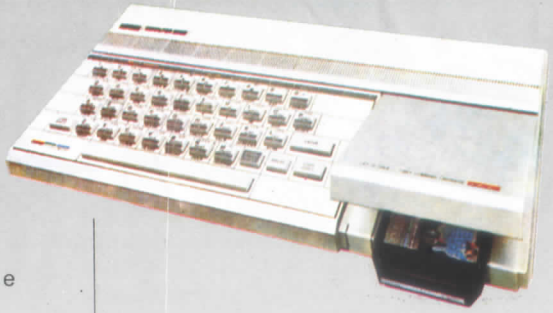
Cód. Postal Cidade

(Em vez de cortar o cupão tire uma fotocópia)

TIMEX COMPUTER **2068**

DOIS COMPUTADORES NUM SÓ...

...CAMPANHA DE TROCAS... UM ANO GARANTIA.



Na linha do ZX/SPECTRUM® e compatível com ele (através de uma cartridge emuladora) apresenta-se mais potente graças à incorporação de um sintetizador de som, um porto para cartridge, dupla resolução gráfica, saída para monitor.

A adição do sistema Floppy Disk Timex permite-lhe explorar novos campos de aplicação onde a realidade ultrapassa a ficção.

Qualquer que seja o seu domínio de actividade encontrará uma aplicação para si!

- Programas profissionais – Contabilidade; Stocks; Tratamento de texto; Base de dados; Gestão de pequenas e médias empresas; Ficheiros personalizados.
- Programas de gestão doméstica/familiar.

CAMPANHA DE TROCAS

O seu Spectrum 48K, Spectrum 16. e Spectrum+, vale (em qualquer estado)

10 MIL ESC.

na aquisição dos TC2048 e TC2068.

Campanha de lançamento, oferta de «joystick» e seis cassettes.

CONSULTE OS REVENDEDORES

TIMEX

NÃO NECESSITA GRAVADOR...

Através do porto de cartridge tem acesso imediato a programas, sem necessidade de recorrer ao tradicional gravador de cassettes.

Cartridges disponíveis: Androids • Crazy Bugs • Budgeter • Flight Simulator • Casino.

Brevemente: Processador de texto • Gestor de leitor de código de barras.

...UMA PORTA ABERTA PARA O FUTURO

ANDROIDS

CRAZYBUGS

BUDGETER

FLIGHT SIMULATOR

CASINO I