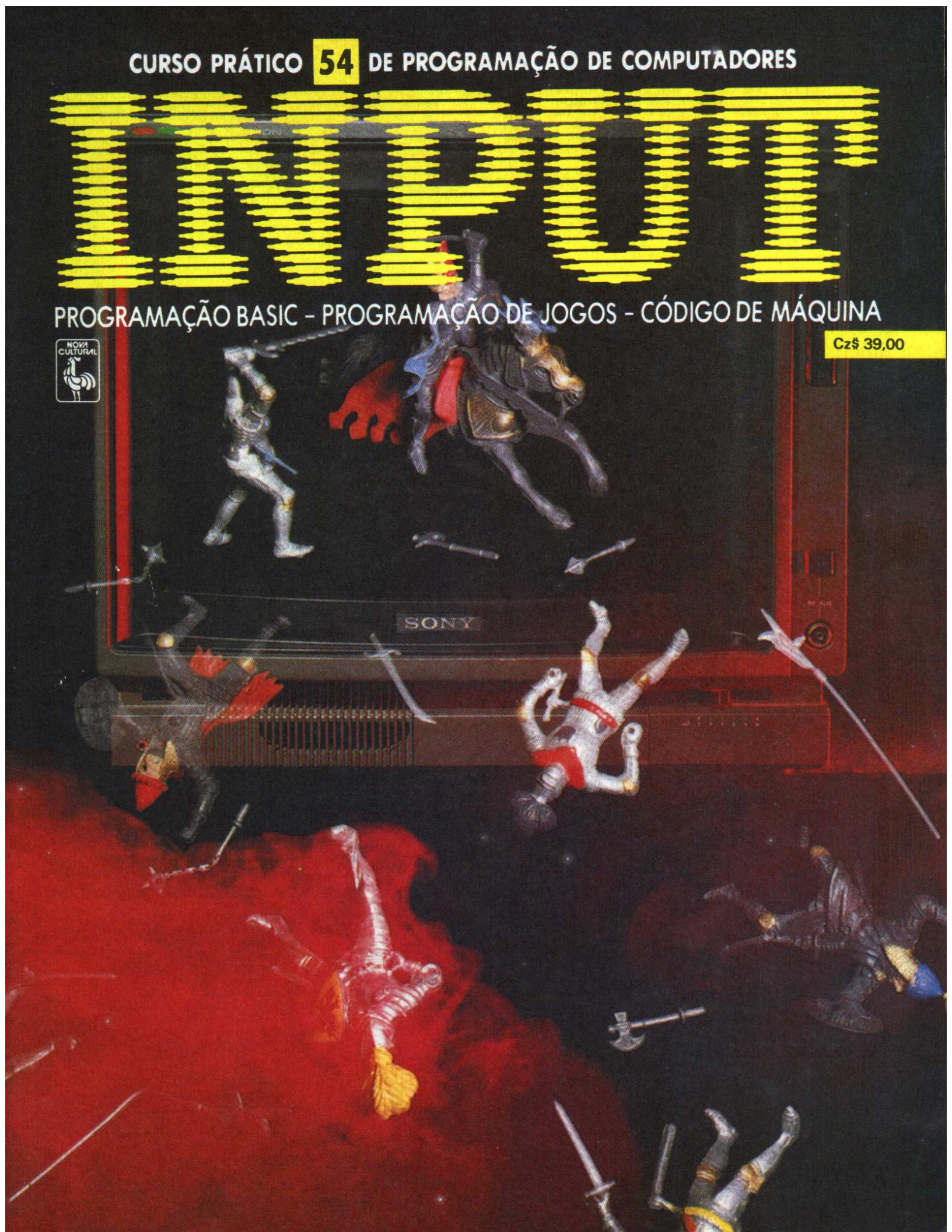
CURSO PRÁTICO 54 DE PROGRAMAÇÃO DE COMPUTADORES

INPUT

PROGRAMAÇÃO BASIC - PROGRAMAÇÃO DE JOGOS - CÓDIGO DE MÁQUINA

NOVA CULTURAL

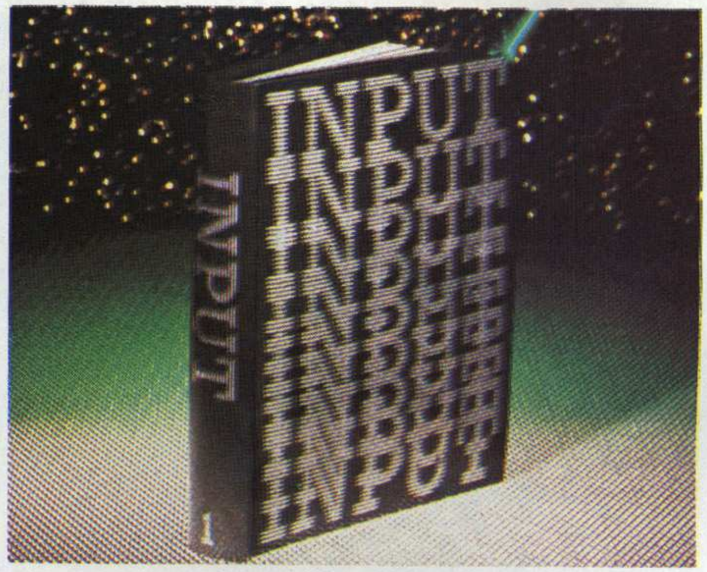
Cz$ 39,00



INPUT

Vol. 4 N.º 54

NESTE NÚMERO



PLANO DA OBRA

"INPUT" é uma obra editada em fascículos semanais, e cada conjunto de 15 fascículos compõe um volume. A capa para encadernação de cada volume estará à venda oportunamente.

COMPLETE SUA COLEÇÃO

Exemplares atrasados, até seis meses após o encerramento da coleção, poderão ser comprados, a preços atualizados, da seguinte forma: 1. PESSOALMENTE — Por meio de seu jornaleiro ou dirigindo-se ao distribuidor local, cujo endereço poderá ser facilmente conseguido junto a qualquer jornaleiro de sua cidade. Em **São Paulo**, os endereços são: rua Brigadeiro Tobias, 773, Centro; avenida Industrial, 117, Santo André; e no **Rio de Janeiro**: avenida Mem de Sá, 191/193, Centro. 2. POR CARTA — Poderão ser solicitados exemplares atrasados também por carta, que deve ser enviada para DINAP — Distribuidora Nacional de Publicações — Números Atrasados — Estrada Velha de Osasco, 132, Jardim Teresa — CEP 06000 — Osasco — SP. Não envie pagamento antecipado. O atendimento será feito pelo reembolso postal e o pagamento, incluindo as despesas postais, deverá ser efetuado ao se retirar a encomenda na agência do Correio. 3. POR TELEX — Utilize o n.º (011) 33 670 DNAP.

Em **Portugal**, os pedidos devem ser feitos à Distribuidora Jardim de Publicações, Lda. — Qta. Pau Varais, Azinhaga de Fetais — 2 685, Camarate — Lisboa; Apartado 57 — Telex 43 069 JARLIS P.

Atenção: Após seis meses do encerramento da coleção, os pedidos serão atendidos dependendo da disponibilidade do estoque.

Obs.: Quando pedir livros, mencione sempre título e/ou autor da obra, além do número da edição.

COLABORE CONOSCO

Encaminhe seus comentários, críticas, sugestões ou reclamações ao Serviço de Atendimento ao Leitor — Caixa Postal 9442, São Paulo — SP.

NOVA CULTURAL

Editor
VICTOR CIVITA

REDAÇÃO
Diretor Editorial: Carmo Chagas

Editores Executivos: Antonio José Filho, Berta Sztark Amar

Editor Chefe: Paulo de Almeida
Editor de Texto: Cláudio A. V. Cavalcanti
Chefe de Arte: Carlos Luiz Batista
Assistentes de Arte: Dagmar Bastos Sampaio, Grace Alonso Arruda, Monica Lenardon Corradi
Secretária de Redação/ Coordenadora: Stefania Crema
Secretários de Redação: Beatriz Hagström, José Benedito de Oliveira Damião, Maria de Lourdes Carvalho, Marisa Soares de Andrade, Mauro de Queiroz

COLABORADORES

Consultor Editorial Responsável: Dr. Renato M. E. Sabbatini (Diretor do Núcleo de Informática Biomédica da Universidade Estadual de Campinas)
Execução Editorial: DATAQUEST Assessoria em Informática Ltda., Campinas, SP
Tradução, adaptação, programação e redação: Abílio Pedro Neto, Aluísio J. Dornellas de Barros, Marcelo R. Pires Therezo, Marcos Huascar Velasco, Raul Neder Porrelli, Ricardo J. P. de Aquino Pereira
Coordenação Geral: Rejane Felizatti Sabbatini
Editora de Texto: Ana Lúcia B. de Lucena

COMERCIAL
Diretor Comercial: Roberto Martins Silveira
Gerente Comercial: Flávio Maculan
Gerente de Circulação: Denise Maria Mozol

PRODUÇÃO
Gerente de Produção: João Stungis
Coordenador de Impressão: Atílio Roberto Bonon

Preparador de Texto/Coordenador: Eliel Silveira Cunha
Preparadores de Texto: Alzira Moreira Braz, Ana Maria Dilguerian, Levon Yacubian, Luciano Tasca, Maria Teresa Galluzzi, Maria Teresa Martins Lopes, Paulo Felipe Mendrone
Revisor/Coordenador: José Maria de Assis
Revisoras: Conceição Aparecida Gabriel, Isabel Leite de Camargo, Ligia Aparecida Ricetto, Maria de Fátima Cardoso, Nair Lucia de Britto
Paste-up: Anastase Potaris, Balduino F. Leite, Edson Donato

Edição organizada pela Editora Nova Cultural Ltda.
Av. Brigadeiro Faria Lima, n.º 2000 - 3.º andar
CEP 01452 - São Paulo - SP - Brasil
(Artigo 15 da Lei 5 988, de 14/12/1973).
Esta obra foi composta na AM Produções Gráficas Ltda.
e impressa na Divisão Gráfica da Editora Abril S.A.

MATEMÁTICA DO CRESCIMENTO

- COMO AS COISAS CRESCEM
- SUPERFÍCIE E VOLUME
- PROBLEMAS ESTRUTURAIS
- RAZÃO DE CRESCIMENTO
- NÚMEROS DE FIBONACCI

Os computadores têm sido utilizados no estudo dos mais diversos aspectos da natureza. Veja aqui como eles podem nos ajudar a compreender o fenômeno do crescimento.

O emprego de computadores na análise das situações relacionadas à nossa realidade não se limita às coisas inanimadas. Ao contrário, a máquina tem se mostrado de grande utilidade no estudo do comportamento dos organismos vivos. Ocorre que a matemática está intimamente associada à natureza — nas formas mais inesperadas. E, sempre que há uma conexão com a matemática, os computadores podem ajudar a compreender o que se passa, efetuando cálculos e indicando resultados.

Neste artigo, examinaremos algumas maneiras de se utilizar computadores na análise do crescimento dos organismos. Todas as formas vivas são capazes, em algum estágio do seu desenvolvimento, de mudar de tamanho, número ou forma. Apenas as duas primeiras situações serão tratadas aqui, juntamente com outros exemplos de inter-relação entre a natureza e a matemática.

COMO AS COISAS CRESCEM

O crescimento — mudança de tamanho — envolve a formação de novos materiais de estrutura. Tanto plantas quanto animais obtêm a matéria-prima necessária para isso do seu meio ambiente. Mas o crescimento geral de um ser pode se dar de duas maneiras: pela multiplicação do número de células (por meio da divisão celular) ou pelo aumento do tamanho das células.

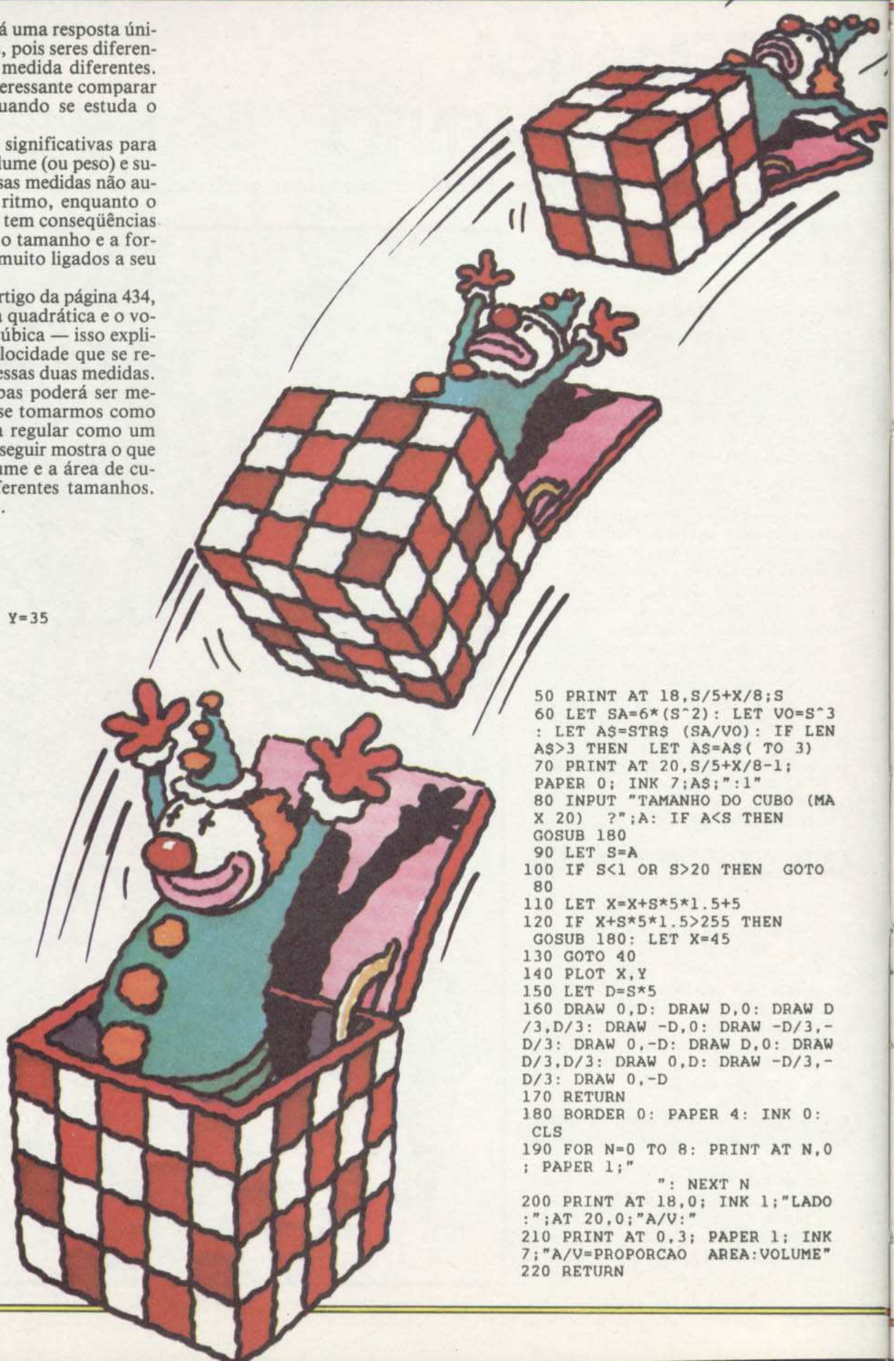
MEDIDAS

Tendo em vista essa diferença, resta saber como medir o crescimento. É melhor medir o peso, como se faz com os bebês, ou a altura, como se faz com as crianças maiores? Ou seria o volume a medida mais adequada? Ou a superfí-

cie do corpo? Não há uma resposta única para tais questões, pois seres diferentes exigem tipos de medida diferentes. Porém, sempre é interessante comparar medidas diversas quando se estuda o crescimento.

As medidas mais significativas para animais são as de volume (ou peso) e superfície corporal. Essas medidas não aumentam no mesmo ritmo, enquanto o animal cresce, o que tem conseqüências importantes, já que o tamanho e a forma de um ser estão muito ligados a seu modo de viver.

Como vimos no artigo da página 434, a área é uma medida quadrática e o volume, uma medida cúbica — isso explica a diferença de velocidade que se registra no aumento dessas duas medidas. A relação entre ambas poderá ser melhor compreendida se tomarmos como exemplo uma forma regular como um cubo. O programa a seguir mostra o que acontece com o volume e a área de cubos que tenham diferentes tamanhos. Digite-o e execute-o.



```
10 GOSUB 180
20 LET X=45: LET Y=35
30 LET S=2
40 GOSUB 140
50 PRINT AT 18,S/5+X/8;S
60 LET SA=6*(S^2): LET VO=S^3
: LET A$=STR$ (SA/VO): IF LEN
A$>3 THEN  LET A$=A$( TO 3)
70 PRINT AT 20,S/5+X/8-1;
PAPER 0; INK 7;A$;":1"
80 INPUT "TAMANHO DO CUBO (MA
X 20)  ?";A: IF A<S THEN
GOSUB 180
90 LET S=A
100 IF S<1 OR S>20 THEN  GOTO
80
110 LET X=X+S*5*1.5+5
120 IF X+S*5*1.5>255 THEN
GOSUB 180: LET X=45
130 GOTO 40
140 PLOT X,Y
150 LET D=S*5
160 DRAW 0,D: DRAW D,0: DRAW D
/3,D/3: DRAW -D,0: DRAW -D/3,-
D/3: DRAW 0,-D: DRAW D,0: DRAW
D/3,D/3: DRAW 0,D: DRAW -D/3,-
D/3: DRAW 0,-D
170 RETURN
180 BORDER 0: PAPER 4: INK 0:
CLS
190 FOR N=0 TO 8: PRINT AT N,0
; PAPER 1;"
                ": NEXT N
200 PRINT AT 18,0; INK 1;"LADO
:";AT 20,0;"A/V:"
210 PRINT AT 0,3; PAPER 1; INK
7;"A/V=PROPORCAO  AREA:VOLUME"
220 RETURN
```

```
10 GOSUB 180:CLS
20 X=45:Y=156
30 S=2
40 GOSUB 140
60 SA=6*S*S:VO=S*S*S
70 PRINT"TAMANHO=";S;TAB(12);"P
ROP.AREA/VOL=";:PRINT USING"#.#
#:1";SA/VO:PRINT
80 INPUT"DIGITE O TAMANHO DO CU
BO (1-20)";A:IF A<S GOSUB 180
90 S=A
100 IF A<1 OR S>20 THEN 80
110 X=X+S*7.5+5
120 IF X+S*7.5>255 GOSUB 180:X=
45
130 GOTO 40
140 SCREEN 1,0
150 DRAW"BM"+STR$(INT(X))+","+S
TR$(INT(Y))+"S"+STR$(INT(S*3))+
"C1E2NR6U6C4R6G2L6NE2D6R6NU6E2U
6"
160 IF INKEY$<>" " THEN 160
170 RETURN
180 PMODE 3:PCLS2
190 RETURN
```

```
10 GOSUB 180:CLS
20 X=45:Y=156
30 S=2
40 GOSUB 140
60 SCREEN0:SA=6*S*S:VO=S*S*S
70 PRINT"TAMANHO=";S;TAB(15);"S
UP/VOL=";:PRINTUSING"#.##:1";SA
/VO:PRINT
80 INPUT"TAMANHO DO CUBO? (1-20
)";A:IF A<S THEN GOSUB 180
90 S=A
100 IF A<1 OR A>20 THEN 80
110 X=X+S*7.5+5
120 IF X+S*7.5>255 THEN GOSUB 1
80:X=45
130 GOTO 40
140 SCREEN2
150 LINE (X,Y)-(X+5*S,Y-5*S),8,
B
160 X=X+2*S:Y=Y-2*S:LINE (X,Y)-
(X+5*S,Y-5*S),8,B
165 LINE (X,Y)-(X-2*S,Y+2*S),8:
LINE (X+5*S,Y)-(X+3*S,Y+2*S),8:
LINE (X+5*S,Y-5*S)-(X+3*S,Y-3*S
),8:LINE (X,Y-5*S)-(X-2*S,Y-3*S
),8:Y=156
170 IF INKEY$="" THEN 170 ELSE
RETURN
180 SCREEN2
190 RETURN
```

```
10  HOME : GOSUB 180
20 X = 45:Y = 156
30 S = 2
40  GOSUB 140
60 SA = 6 * S * S:VO = S * S *
S
70  VTAB 21: CALL  - 958: PRINT
 "TAMANHO = ";S: PRINT "RAZAO S
UP/VOL = ";SA / VO;":1"
80  INPUT "TAMANHO DO CUBO (1-2
0)? ";A: IF A < S THEN  GOSUB 1
80
90 S = A
100  IF A < 1 OR S > 20 THEN 80

110 X = X + S * 7.5 + 5
120  IF X + S * 7.5 > 255 THEN
 GOSUB 180:X = 45
130  GOTO 40
140  HPLOT X,Y TO X + 5 * S,Y T
O X + S * 5,Y - S * 5 TO X,Y -
5 * S TO X,Y
150 U =  ABS (U - 1): IF U THEN
 X = X + 2 * S:Y = Y - 2 * S: G
OTO 140
160  HPLOT X,Y TO X - 2 * S,Y +
 2 * S: HPLOT X + 5 * S,Y TO X
+ 3 * S,Y + 2 * S: HPLOT X + 5
* S,Y - 5 * S TO X + 3 * S,Y -
3 * S: HPLOT X,Y - 5 * S TO X -
 2 * S,Y - 3 * S
170 Y = 156: RETURN
180  HGR : HCOLOR= 3: RETURN
```

O programa pede ao usuário que escolha um número para a aresta (a medida de uma quina à outra) do cubo. Em seguida, desenha a figura na tela e mostra a razão entre área e volume. Comece com um cubo pequeno — quatro unidades de aresta, por exemplo — e vá aumentando o valor fornecido ao computador para simular um crescimento. Você verá que, à medida que o cubo se torna maior, a razão entre área e volume diminui. Em outras palavras, o volume aumenta muito mais rapidamente que a superfície do corpo.

Grande parte da listagem está destinada a formatar a tela e desenhar os cubos. Assim, vamos nos deter na linha 60, cujo conteúdo nos interessa de maneira mais direta.

S é a aresta do cubo. A área de um lado é, portanto, S*S e a superfície do cubo todo (ele tem seis lados) é 6*S*S. O volume é S ao cubo, ou S*S*S. A razão entre as duas medidas corresponde ao resultado da divisão da superfície pelo volume. Por exemplo, quando S é igual a 2, a razão é 6*2*2 dividido por 2*2*2, que dá 3:1. Se dobramos o tamanho da aresta, temos 6*4*4 dividido por 4*4*4, que dá 1,5:1 — ou seja, a área é relativamente menor.

Como animais não têm forma de cubo, é provável que queira adaptar o programa para torná-lo mais próximo da realidade. Pequenos animais, como os camundongos, são melhor representados por uma esfera, enquanto seres humanos lembram mais um conjunto de cilindros — um para o tronco e outro para cada membro. No primeiro caso, precisamos conhecer a superfície e o volume da esfera. A superfície é **4*PI*raio^2** e o volume, **4/3*PI*raio^3**. Já a área de um cilindro é **2*PI*raio*altura** mais **2*PI*raio^2** e o volume, **PI*raio^2*altura**.

Seja qual for a forma escolhida, o princípio geral é o mesmo e, em todos os casos, o volume cresce mais rapidamente que a superfície.

CRESCIMENTO E VIDA

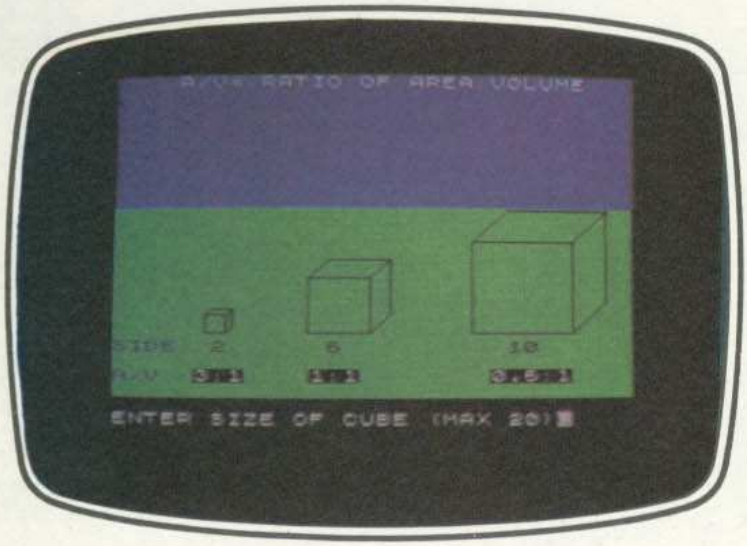
Essas relações matemáticas são muito importantes no Reino Animal, afetando as atividades e o habitat do ser em questão. Tomemos como exemplo o rato: ele tem um volume pequeno e, portanto, uma área relativa grande. Isso significa que seu corpo irradia calor rapidamente (dependendo do ambiente em que se encontra). O rato consegue manter-se aquecido produzindo energia pela queima de alimento. Ora, graças à velocidade com que perde calor, ele precisa ingerir alimentos em quantidade equivalente à metade do seu peso.

Um elefante, em contrapartida, tem um volume muito grande em comparação com sua superfície corporal. Portanto, mantidas as proporções, ele irradia menos calor que o rato, e sobrevive com uma quantidade menor (em relação ao seu peso) de alimentos diários. Isso explica por que animais grandes se adaptam melhor ao clima frio dos pólos, onde a comida é escassa.

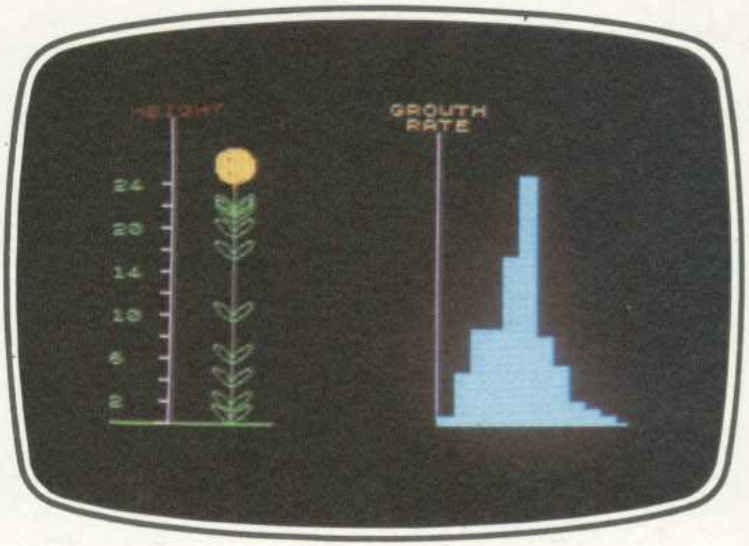
LIMITE DE TAMANHO

A relação entre massa e superfície ajuda a explicar por que animais e plantas não crescem além de certo tamanho. Dadas as proporções de uma determinada espécie, há um limite definido para as dimensões que ela pode alcançar. Esse limite é imposto pelo peso que os ossos do animal podem suportar.

Quando seu tamanho — isto é, altura, comprimento e largura — dobra, o peso cresce oito vezes, mas a área de suporte dos ossos cresce apenas quatro vezes. Se os ossos se desenvolvessem o suficiente para sustentar o novo peso, ficariam desproporcionalmente finos em relação ao tamanho do animal. Em determinado ponto do seu crescimento, o animal estaria tão desajeitado que não conseguiria se mover.



Volumes e superfícies: gráfico comparativo.



Alterações da taxa de crescimento de uma planta.

Se um rato atingisse o tamanho de um elefante, mantidas as mesmas proporções iniciais, suas pernas seriam incapazes de suportá-lo. Isso explica por que espécies de tamanhos diversos têm proporções tão diferentes.

O maior animal terrestre — o elefante — chega a pesar cerca de 10 toneladas. Um dinossauro podia alcançar 80 toneladas, mas ficava a maior parte do tempo imerso na água. Os animais marinhos contam com o suporte da água, o que amplia os limites de seu tamanho. Baleias azuis têm peso superior a 150 toneladas e medem mais de 30 metros de comprimento. Naturalmente, não haveria pernas que fossem capazes de sustentá-las fora da água. O limite de seu crescimento, assim como dos demais animais marinhos, está mais relacionado à perda de calor.

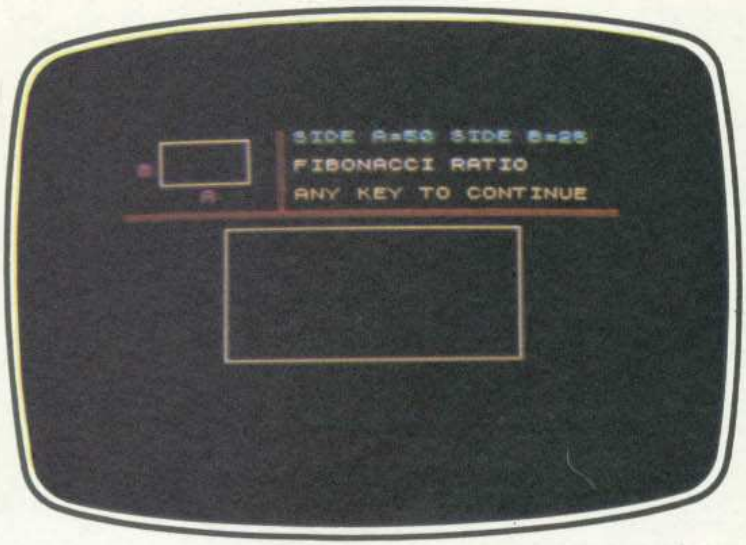
RAZÃO DE CRESCIMENTO

O programa a seguir mostra a velocidade de crescimento de uma planta, do plantio à maturidade, quando ela chega a seu limite de tamanho.

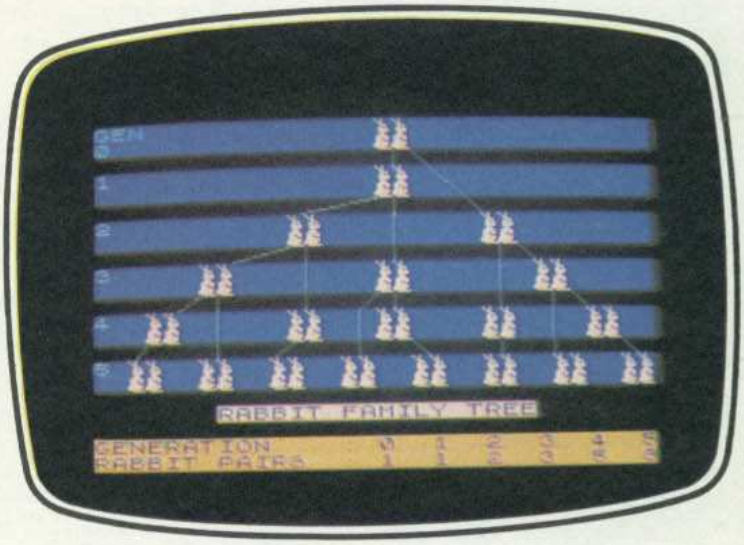
A planta cresce devagar no início, e rapidamente depois. Atingindo seu limite de tamanho, volta a se desenvolver bem devagar, até parar ou, eventualmente, morrer. Digite e execute o seguinte programa:

```
10 DIM G(11)
20 DATA 2,9,22,35,58,92,104,
112,115,117,118
30 FOR N=1 TO 11: READ G(N):
NEXT N
40 BORDER 0: PAPER 0: INK 4:
CLS
50 LET X=60: LET Y=10
60 DRAW 80,0
70 PLOT 30,0: DRAW  INK 7;0,
168
80 PRINT  INK 2;AT 0,1;"ALTUR
A"
90 FOR N=0 TO 138 STEP 12
100 PLOT  INK 7;30,N
110 DRAW  INK 7;-5,0
120 NEXT N
130 PRINT AT 20,0;"2";AT 17,0;
"6";AT 14,0;"10";AT 11,0;"14";
AT 8,0;"20";AT 5,0;"24"
140 PLOT 160,0: DRAW  INK 7;0,
168
150 PLOT 160,0: DRAW  INK 7;95
,0
160 PRINT AT 0,17; INK 6;"TAXA
 DE";AT 1,15;"CRESCIMENTO"
170 LET C=1
180 LET GX=161
190 LET GY=1
200 FOR N=1 TO 117
210 IF G(C)<>N THEN  GOTO 250
220 PLOT X,Y: DRAW 9,9,PI/2:
DRAW -9,-9,PI/2
230 DRAW -9,9,PI/2: DRAW 9,-9,
PI/2
240 LET GX=GX+8: LET GY=1: LET
C=C+1
250 PLOT X,Y
260 FOR K=0 TO 3
270 PLOT  INK 5;GX,GY+K: DRAW
INK 5;8,0
280 NEXT K
290 LET GY=GY+4
300 LET Y=Y+1
310 NEXT N
320 FOR Y=117 TO 140
330 PLOT X,Y
340 NEXT Y
350 FOR R=1 TO 10 STEP .3
360 CIRCLE  INK 6;X,Y,R
370 NEXT R
380 GOTO 380
```

```
20 DATA 2,9,22,35,58,92,104,112
,115,117,118
30 FOR N=0 TO 10:READ G(N):NEXT
40 PMODE 3:PCLS:SCREEN 1,1
50 X=60:Y=190
60 LINE (8,23)-(8,192),PSET:LIN
E-(80,192),PSET
80 DRAW"BM20,6S24C7D2BRUNLUBR2L
DNRDRBRU2BR2LD2RUS8NLS24BED2BRU
NLUBRS16RND3RC8"
90 FOR N=47 TO 191 STEP 12
100 LINE (2,N)-(8,N),PSET
120 NEXT
140 LINE (158,23)-(158,191),PSE
T
150 LINE -(255,191),PSET
160 DRAW"BM176,6C6S24LD2RUS8NLS
24BEND2RDLFBRNU2RU2LBR2D2EFU2BR
S16RND3RS24BRD2BRUNLUBM182,24ND
2RDLFBRU2RDNLDS16BR2U3LR2S24BR2
LDNRDR"
170 GX=161:GY=190
190 COLOR 6,7
200 FOR N=1 TO 117
210 IF G(C)>N THEN 250
220 CIRCLE(X,Y-4),15,6,.4,0,.5
230 CIRCLE(X-16,Y),16,6,.4,.75,
1:CIRCLE(X+16,Y),16,6,.4,.5,.75
240 GX=GX+8:GY=190:C=C+1
250 PSET(X,Y,6)
260 FOR K=0 TO 3
270 LINE(GX,GY-K)-(GX+8,GY-K),P
RESET
280 NEXT
290 GY=GY-4
300 Y=Y-1
310 NEXT
320 FOR Y=75 TO 58 STEP -1
330 PSET (X,Y,6)
340 NEXT:POKE 178,54
350 FOR R=1 TO 15
360 CIRCLE (X,Y),R,.8
370 NEXT
380 GOTO 380
```



Regra de Fibonacci para um retângulo bem proporcionado.



Uma explosão populacional de coelhos.

```
20 DATA 2,9,22,35,58,92,104,112
,115,117,118
30 FOR N=1 TO 10:READ G(N):NEXT
40 COLOR 2,15,15:SCREEN2
50 X=60:Y=190:PI=3.1416
60 LINE (8,23)-(8,192):LINE -(8
0,192)
90 FOR N=47 TO 191 STEP 12
100 LINE (2,N)-(8,N)
120 NEXT
140 LINE (158,23)-(158,191),8
150 LINE -(255,191),8
170 GX=161:GY=190
200 FOR N=1 TO 117
210 IF G(C)>N THEN 250
220 LINE(X,Y)-(X+10,Y-5):LINE(X
,Y)-(X-10,Y-5)
240 GX=GX+8:GY=190:C=C+1
250 PSET(X,Y)
260 FOR K=1 TO 3
270 LINE(GX,GY-K)-(GX+8,GY-K),8
280 NEXT
290 GY=GY-4
300 Y=Y-1
310 NEXT
320 FOR Y=75 TO 58 STEP-1
330 PSET(X,Y),6
340 NEXT
350 FOR R=1 TO 14
360 CIRCLE(X,Y),R,16-R,,,.8
370 NEXT
380 GOTO 380
```

```
20  DATA  2,9,22,35,58,92,104,1
12,115,117,118
30  FOR N = 1 TO 10: READ G(N):
 NEXT
40  HOME : HGR : HCOLOR= 3
50 X = 60:Y = 156
60  HPLOT 8,2 TO 8,158 TO 80,15
8
80  VTAB 21: PRINT "  TAM DA PL
ANTA","    VEL DE CRESCIMENTO"
90  FOR N = 11 TO 155 STEP 12
100  HPLOT 2,N TO 8,N
120  NEXT
140  HPLOT 158,2 TO 158,156 TO
255,156
170 GX = 161:GY = 156
200  FOR N = 1 TO 117
210  IF G(C) > N THEN 250
230  HPLOT X,Y TO X + 10,Y - 5:
 HPLOT X,Y TO X - 10,Y - 5
240 GX = GX + 8:GY = 156:C = C
+ 1
250  HPLOT X,Y
260  FOR K = 1 TO 3
270  HPLOT GX,GY - K TO GX + 8,
GY - K
280  NEXT
290 GY = GY - 4
300 Y = Y - 1
310  NEXT
340 CX = X:CY = Y - 15:R1 = 13:
R2 = 8: HPLOT CX + R,CY
350  FOR A = 0 TO 6.28 STEP .1
360  HPLOT  TO CX + R1 *  COS (
A),CY - R1 *  SIN (A): HPLOT  T
O CX + R2 *  COS (A),CY - R2 *
 SIN (A)
370  NEXT
```

O programa mostra graficamente o que acontece com a planta. Os dados da linha 20 equivalem ao tamanho da planta, medido a intervalos regulares. São usados tanto no desenho da figura quanto no traçado do gráfico que mostra a velocidade de crescimento.

A rotina da linha 200 à 310 encarrega-se do desenho da planta. Os eixos e escalas do gráfico são definidos nas linhas 40 a 160, enquanto o traçado das barras é feito pelas linhas 260 a 280. A flor, finalmente, é desenhada pelas linhas 350 a 380.

Como o gráfico deixa claro, o crescimento da planta é lento no princípio, acelerando em seguida, até perder, de novo, a velocidade. Os dados foram retirados de um experimento real que mediu o crescimento da área de uma folha de um pepineiro. Os mesmos valores podem ser usados para avaliar o crescimento de toda a planta.

GERAÇÕES E NÚMEROS

Quando se acasalam, os animais multiplicam-se, dando origem a várias gerações. Tomemos o caso dos coelhos. Um par de coelhos — primeira geração — produz um novo par. Este forma a segunda geração. O casal inicial produz mais um par e a segunda geração passa a ter dois pares. Como veremos adiante, uma série de números representando gerações pode ser como se segue: 1,1,2,3,5,8,13,21 etc. O próximo programa mostra graficamente como o número de coelhos cresce com o passar do tempo.

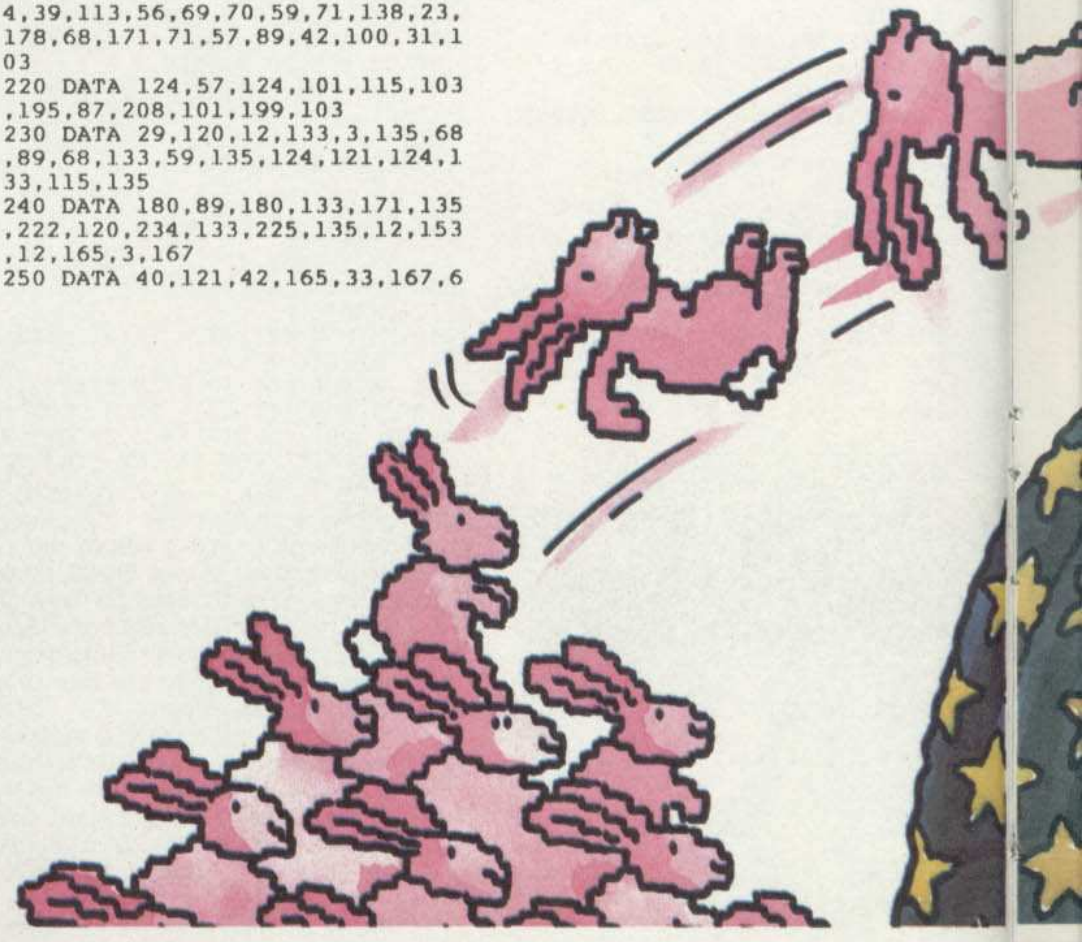
```
10 BORDER 0: PAPER 1: INK 7:
CLS
20 FOR N=0 TO 7: READ A: POKE
USR "a"+N,A: NEXT N
30 FOR N=0 TO 7: READ A: POKE
USR "b"+N,A: NEXT N
40 LET C$="
                    "
50 LET A$=CHR$ 144: LET B$=
CHR$ 145
60 PAPER 0: CLS : PAPER 1
70 FOR N=1 TO 6: PRINT C$'':
NEXT N
80 PRINT  INK 5;AT 0,0;"GER"
```

```
"''"0"''"1"''"2"''"3"''"4"''"5"
 90 FOR N=1 TO 20
100 IF N>1 THEN  FOR P=1 TO 10
: SOUND .01,P: NEXT P
110 READ X,Y: PRINT AT Y,X;A$;
A$;AT Y+1,X;B$;B$
120 NEXT N
130 FOR N=1 TO 20
140 READ X,Y,XX,YY: PLOT X,Y:
DRAW  INK 4;XX,YY
150 NEXT N
160 PRINT AT 18,2; INVERSE 1;"
ARVORE GENEALOGICA - COELHOS"
170 INK 6: INVERSE 1: PRINT '"
GERACAO       : 0  1  2  3  4
 5"''"PARES         : 1  1  2
3  5  8"
180 GOTO 180
190 DATA 144,80,48,28,52,62,62
,24,60,126,118,120,126,254,252
,63
200 DATA 16,0,16,3,22,6,11,6,
16,9,6,9,11,12,25,9,22,12,3,12
,6,15,14,15,16,12,28,12,26,15,
2,15,10,15,18,15,22,15,30,15
210 DATA 136,159,0,-7,144,159,
32,-31,128,135,-34,-7,136,135,
0,-31,184,112,0,-31,192,112,8,
-8,88,111,-25,-7,96,111,0,-31
220 DATA 48,87,-10,-7,56,87,0,
-31,24,63,-5,-7,90,63,-8,-7,
128,87,-8,-8,120,79,0,-23,136,
87,0,-7
230 DATA 144,63,7,-7,184,63,0,
-7,208,88,0,-31,216,88,7,-7,
240,63,7,-7
```

```
10 PMODE 3:PCLS:DIM R(9)
20 SS=PEEK(186)*256+PEEK(187)
30 FOR K=SS TO SS+480 STEP 32
40 READ A,B:POKE K,A:POKE K+1,B
50 NEXT
60 GET (2,0)-(13,15),R,G
70 PUT(14,0)-(25,15),R,PSET:GET
(2,0)-(25,15),R,G
80 PCLS4:SCREEN 1,0
90 COLOR3: FOR K=0 TO 5
100 LINE (0,32*K)-(255,32*K+24)
,PSET,BF
110 NEXT
120 COLOR 1:FOR N=1 TO 20
140 READ X,Y:PUT(X,Y)-(X+23,Y+1
5),R,PSET
150 IF N>1 THEN PLAY"O1T50CCEFG
AB"
160 READ X,Y,XX,YY:LINE(X,Y)-(X
X,YY),PSET
170 NEXT
180 GOTO 180
190 DATA 169,170,153,170,165,17
0,169,90,165,154,165,86,15,90,1
69,106
200 DATA 169,106,165,90,165,154
,165,106,165,106,149,106,149,10
6,165,90
210 DATA 114,7,124,24,124,38,11
4,39,113,56,69,70,59,71,138,23,
178,68,171,71,57,89,42,100,31,1
03
220 DATA 124,57,124,101,115,103
,195,87,208,101,199,103
230 DATA 29,120,12,133,3,135,68
,89,68,133,59,135,124,121,124,1
33,115,135
240 DATA 180,89,180,133,171,135
,222,120,234,133,225,135,12,153
,12,165,3,167
250 DATA 40,121,42,165,33,167,6
8,153,72,165,63,167,113,120,98,
165,93,167,124,153,144,165,137,
167
260 DATA 178,153,176,165,167,16
7,208,121,206,165,197,167,234,1
53,234,165,225,167,124,70,124,7
0
```

```
5 CLEAR 5000
10 COLOR 4,15,15:SCREEN2
60 A$="S4F1D1L2BF1R3G1L1D1R1F1L
3D1R3"
90 FOR K=0 TO 5
100 LINE(0,32*K)-(255,32*K+24),
2,BF
110 NEXT
120 FOR N=1 TO 20
140 READ X,Y:PRESET(X,Y):DRAW"X
A$;":PRESET(X+15,Y):DRAW"XA$;"
160 READ X,Y,XX,YY:LINE(X,Y)-(X
X,YY),8
170 NEXT
180 GOTO 180
210 DATA 114,7,124,24,124,38,11
4,39,113,56,69,70,59,71,138,23,
178,68,171,71,57,89,42,100,31,1
03
220 DATA 124,57,124,101,115,103
,195,87,208,101,199,103
230 DATA 29,120,12,133,3,135,68
,89,68,133,59,135,124,121,124,1
33,115,135
240 DATA 180,89,180,133,171,135
,222,120,234,133,225,135,12,153
,12,165,3,167
250 DATA 40,121,42,165,33,167,6
8,153,72,165,63,167,113,120,98,
165,93,167,124,153,144,165,137,
167
260 DATA 178,153,176,165,167,16
7,208,121,206,165,197,167,234,1
53,234,165,225,167,124,70,124,7
0
```

```
10  HGR2 : HCOLOR= 3
20  DATA  1,0,4,0,60,60,28,109,
1,193,193,43,53,55,53,54,55,45,
53,63,63,46,173,27,54,45,45,62,
63,63,46,45,53,63,55,45,53,63,6
2,63,60,7,0
30 P = 233: POKE P - 1,0: POKE
P,97: SCALE= 1: ROT= 0
40  FOR I = 24832 TO 24874: REA
D C: POKE I,C: NEXT
90  FOR K = 0 TO 5
100  HPLOT 0,30 * K TO 255,30 *
 K
110  NEXT
120  FOR N = 1 TO 20
140  READ X,Y: DRAW 1 AT X,Y: D
RAW 1 AT X + 15,Y
160  READ X,Y,XX,YY: HPLOT X,Y
TO XX,YY
170  NEXT
190  DATA   114,7,124,24,124,38
,114,39,113,56,69,70,59,71,138,
```



```
23,178,68,171,71,57,89,42,100,3
1,103
200 DATA 124,57,124,101,115,1
03,195,87,208,101,199,103
210 DATA 29,120,12,133,3,135
,68,69,68,133,59,135,124,121,12
4,133,115,135
220 DATA 180,89,180,133,171,
135,222,120,234,133,225,135,12,
153,12,165,3,167
230 DATA 40,121,42,165,33,167
,68,153,72,165,63,167,113,120,9
8,165,93,167,124,153,144,165,13
7,167
240 DATA 178,153,176,165,167
,167,208,121,206,165,197,167,23
4,153,234,165,225,167,124,70,12
4,70
```

O programa começa criando o UDG para o coelho, a partir das linhas de dados (190 e 200). As linhas 60 a 110 (40 a 80, no Spectrum) desenham barras na tela para separar cada geração. A seção seguinte do programa, que vai até a linha 180, imprime os coelhos e as linhas que ligam as gerações. Os dados que definem as linhas e posições são lidos da linha 210 em diante.

À medida que os organismos se multiplicam, eles se espalham e colonizam outras áreas. Um bom exemplo é a maneira pela qual as bactérias se reproduzem. Computadores também podem ser usados para representar esse tipo de expansão. Existem jogos interessantes elaborados a partir dessa idéia. Você encontrará um deles, o *Jogo da Vida*, na página 961.

NÚMEROS DE FIBONACCI

A série de números mencionada anteriormente — 1,1,2,3,5,8,13,21 etc. — apresenta algumas propriedades que podem ser observadas na natureza e em obras de arte. É conhecida como "números de Fibonacci" desde o século XIII, quando o matemático italiano a descreveu. Cada número dessa série corresponde à adição dos dois anteriores.

Uma propriedade interessante pode ser observada tomando-se quaisquer três números em seqüência. Multiplique o primeiro pelo último e compare o resultado com o quadrado do número do meio. A diferença será sempre 1. Tomemos os números 5, 8 e 13: 5 vezes 13 dá 65 e 8 ao quadrado, 64.

Dividindo cada número pelo da direita, obtemos uma série de frações que se relacionam à natureza e à arte. Descobriu-se, por exemplo, que nem todas as formas retangulares são igualmente agradáveis de se olhar. Algumas parecem muito estreitas, outras muito largas. O retângulo de melhor aparência tem uma relação especial entre altura e largura, conhecida como a "razão de ouro". Essa razão é igual a (sqr (5) – 1)/2, cujo resultado é 0.6180. Se você calcular qualquer das frações de Fibonacci, verá que, quanto maiores os números usados, mais elas se aproximam da razão de ouro. Por exemplo, 8/13 é igual a 0,6154; 13/21, a 0,6190 e 21/34, a 0,6176.

Experimente agora o próximo programa. Ele desenha retângulos de diferentes tamanhos, de maneira que você mesmo poderá julgar quais os melhores.

```
 10 DIM F(12): DIM D(14)
 20 LET D(1)=1: LET D(2)=1
 30 FOR N=3 TO 14
 40 LET D(N)=D(N-1)+D(N-2)
 50 NEXT N
 60 FOR N=1 TO 12
 70 LET F(N)=D(N)/D(N+1)
 80 NEXT N
 90 BORDER 0: INK 7: PAPER 0:
 CLS
100 LET A=15: LET B=8
110 LET X=20: LET Y=170
120 GOSUB 310
130 PLOT 0,130: DRAW  INK 2;
255,0
140 PLOT 0,128: DRAW  INK 2;
255,0
150 PLOT 80,130: DRAW  INK 2;0
,45
160 PLOT 82,130: DRAW  INK 2;0
,45
170 PRINT AT 2,1; INK 3;"B";AT
4,5;"A"
180 INPUT "COMPRIMENTO DO LADO
 A (MAX 70) ?";A
190 IF A<1 OR A>70 THEN  GOTO
 180
200 INPUT "COMPRIMENTO DO LADO
 B (MAX 40) ?";B
210 IF B<1 OR B>40 THEN  GOTO
 200
220 LET X=128-(A*3/2): LET Y=
 120
230 GOSUB 310
240 PRINT AT 0,11; INK 5;"LADO
 A=";A;" LADO B=";B
250 FOR N=1 TO 12
260 IF A/B=F(N) OR B/A=F(N)
 THEN  PRINT AT 2,8;"RAZAO DE
FIBONACCI"
270 NEXT N
280 PRINT AT 4,1; FLASH 1; INK
6;"QUALQUER TECLA PARA CONTINU
AR"
290 PAUSE 0
300 RUN
310 PLOT X,Y: DRAW 3*A,0: DRAW
0,-3*B
320 DRAW -3*A,0: DRAW 0,3*B
330 RETURN
```

```
10 DIM F(11),D(13)
20 D(0)=1:D(1)=1
30 FOR N=2 TO 13
40 D(N)=D(N-1)+D(N-2)
50 NEXT
60 FOR N=0 TO 11
70 F(N)=D(N)/D(N+1)
80 NEXT
90 PMODE 3:PCLS:CLS
100 A=15:B=8
110 X=20:Y=22
120 GOSUB 310
130 COLOR 4:LINE(0,62)-(255,62)
,PSET
140 LINE(0,64)-(255,64),PSET
150 LINE(80,62)-(80,17),PSET
160 LINE(82,62)-(82,17),PSET
170 DRAW"BM9,38C2S8U4R2FGNLFGL2
BM40,58U3EFDNLD2"
175 FOR K=1 TO 900:NEXT
180 INPUT"COMPRIMENTO DO LADO A
 (MAX 70) ";A
190 IF A<1 OR A>70 THEN 180
200 INPUT"COMPRIMENTO DO LADO B
 (MAX 40) ";B
210 IF B<1 OR B>40 THEN 200
220 X=128-A*S/3:Y=72
230 GOSUB 310
250 FOR N=0 TO 11
260 IF A/B=F(N) OR B/A=F(N) THE
N DRAW"BM106,44C2S8NR2D2NRD2BR4
U4BR2ND4RFGNLFGLBR4NU4R2U4L2BR4
DND3F2NU3DBR2U3EFDNL2D2BR4L2U4R
2BR4L2D4R2BR2U4BR8ND4R2D2L2F2BR
2U3EFDNLD2BR3U4LR2BR2D4BR2R2U4L
2D4"
270 NEXT
280 IF INKEY$="" THEN 280
300 RUN
310 SCREEN 1,0:COLOR 3
320 LINE(X,Y)-(3*A+X,Y+3*B),PSE
T,BF
330 RETURN
```

```
10 DIM F(11),D(13)
20 D(0)=1:D(1)=1
30 FOR N=2 TO 13
40 D(N)=D(N-1)+D(N-2)
50 NEXT
60 FOR N=0 TO 11
70 F(N)=D(N)/D(N+1)
80 NEXT:OPEN "GRP:" FOR OUTPUT
AS #1
85 COLOR  15,2,2:GOTO 180
90 SCREEN2
100 A=15:B=8
110 X=20:Y=22
120 GOSUB 310
130 LINE(0,62)-(255,62),8
140 LINE(0,64)-(255,64),8
150 LINE(80,62)-(80,17),8
160 LINE(82,62)-(82,17),8
170 FOR K=1 TO 1500:NEXT:RETURN
180 SCREEN0:INPUT"TAMANHO DO LA
DO A (MAX 70)";AA
190 IF AA<1 OR AA>70 THEN 180
200 INPUT"TAMANHO DO LADO B (MA
X 40)";BB
210 IF BB<1 OR BB>40 THEN 200
215 GOSUB 90
220 A=AA:B=BB:X=128-A*3/2:Y=72
230 GOSUB 310
240 FOR N=0 TO 11
260 IF A/B=F(N) OR B/A=F(N) THE
N PRESET(90,15):PRINT#1,"RAZAO
DE FIBONACCI"
270 NEXT
280 IF INKEY$="" THEN 280
300 RUN
310 REM
320 LINE(X,Y)-(3*A+X,Y+3*B),4,B
F
330 RETURN
```

```
10  DIM F(11),D(13)
20 D(0) = 1:D(1) = 1
30  FOR N = 2 TO 13
40 D(N) = D(N - 1) + D(N - 2)
50  NEXT
60  FOR N = 0 TO 11
70 F(N) = D(N) / D(N + 1)
80  NEXT
90  HOME : HGR : HCOLOR= 3
100 A = 15:B = 8
110 X = 20:Y = 22
120  GOSUB 310
130  HPLOT 0,62 TO 255,62
140  HPLOT 0,64 TO 255,64
150  HPLOT 80,62 TO 80,17
160  HPLOT 82,62 TO 82,17
180  VTAB 21: INPUT "TAMANHO DO
 LADO A (MAX 70) ";A
190  IF A < 1 OR A > 70 THEN 18
0
200  INPUT "TAMANHO DO LADO B (
MAX 39) ";B
210  IF B < 1 OR B > 39 THEN 20
0
220 X = 128 - A * 3 / 2:Y = 72
230  GOSUB 310
250  FOR N = 0 TO 11
260  IF A / B = F(N) OR B / A =
 F(N) THEN  VTAB 21: CALL  - 95
8: PRINT " RAZAO DE FIBONACCI "
; CHR$ (7)
270  NEXT
280  FOR I = 1 TO 1000: NEXT :
POKE  - 16302,0: GET R$
300  GOTO 90
310  FOR XX = X TO X + 3 * A
320  HPLOT XX,Y TO XX,Y + 3 * B

325  NEXT
330  RETURN
```

O programa começa pedindo que o usuário defina o tamanho dos lados do retângulo. Se você entrar dois números adjacentes da série de Fibonacci, será avisado de que se trata de uma fração de Fibonacci.

Os números da série são calculados nas linhas que vão de 20 a 50. A partir dos dois primeiros números, os demais vão sendo calculados pela adição de cada número ao anterior. As linhas 60 a 80 guardam os números em uma matriz e as linhas 90 a 170 desenham um retângulo, para exemplo. Em seguida, começa a rotina de entrada, que verifica se os números escolhidos pertencem à série de Fibonacci.

Podem-se encontrar exemplos dos números de Fibonacci também na natureza; assim, uma espiral ligando folhas de um galho tem voltas e vãos que formam razões de Fibonacci. Conte o número de voltas da espiral, de uma folha à outra. Depois, conte o número de vãos da espiral entre essas duas posições. A razão é, em geral, 5/3 ou 8/5.

JOGOS DE GUERRA: ÀS ARMAS

- ALCANCE DOS PROJÉTEIS
- COMBATE CORPO A CORPO
- CONFERINDO O MORAL
- VITÓRIA E DERROTA
- INSTRUÇÕES

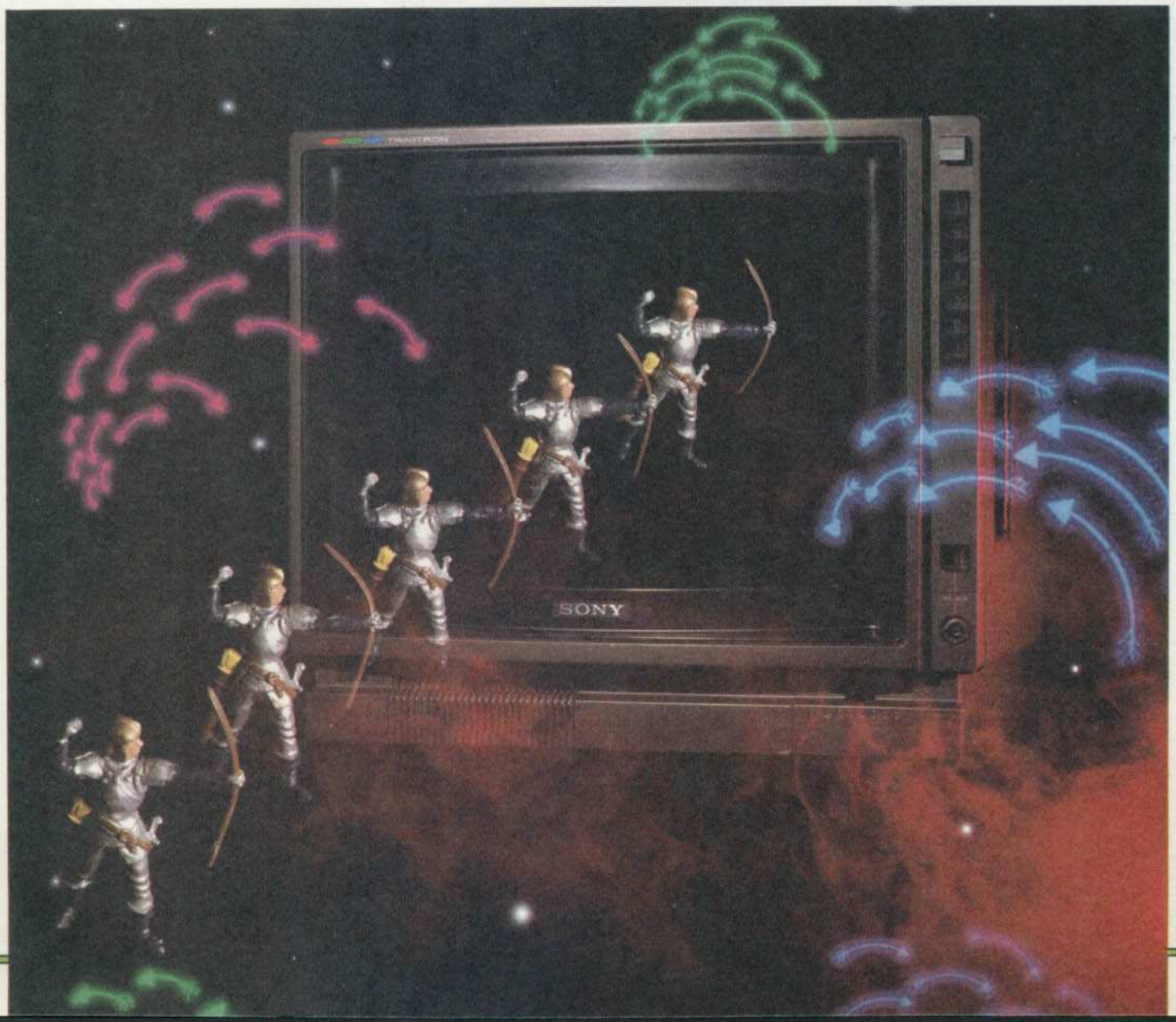
A batalha começa, afinal, possibilitada pelas rotinas de combate balístico e corpo a corpo que apresentamos neste artigo. Veremos também como testar o moral e contar as baixas de cada lado.

Capa e Espada está quase completo; já podemos dar ordens às unidades e movê-las pelo campo de batalha. Faltam apenas as rotinas de combate. Depois de adicioná-las ao programa, experimentaremos, finalmente, o jogo.

Os eventos resultantes do choque entre dois exércitos podem ser bem complicados — assim, antes de qualquer coisa, devemos decidir o que incluir no programa. No tipo mais simples de resolução de combate, o maior sempre vence, o que faz do tamanho das unidades o fator decisivo da vitória ou da derrota. Podemos ainda relacionar o desenlace da batalha ao moral da tropa, ao número de cavaleiros etc. Na realidade, ninguém tem a receita infalível do triunfo; portanto, os fatores que determinam quais serão os vencedores em um jogo de guerra dependem da escolha do programador.

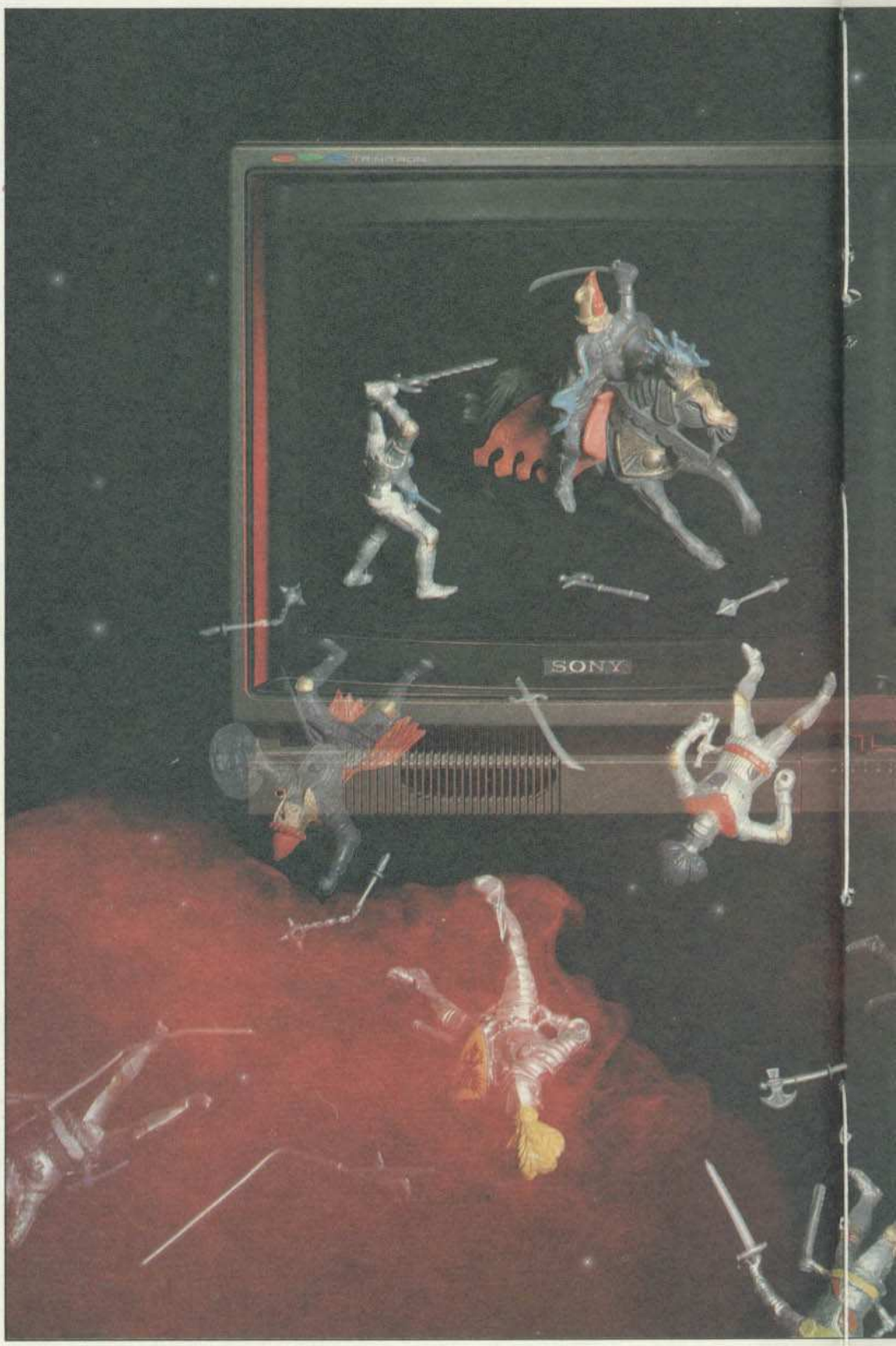
COMBATE BALÍSTICO

Em *Capa e Espada* existem dois tipos de combate: balístico (flechas) e corpo a corpo. Esta primeira rotina trata do combate com projéteis.



```
1710 REM  Tiro
1720 GOSUB 2540
1730 PRINT AT 18,0;"Unidade ";s
h;" atira"
1740 LET fx=5: LET fy=5: LET gp
=-1
1745 LET st=9
1750 IF sh>8 THEN  LET st=1
1770 FOR m=st TO (st+7)
1780 LET tm=ABS (T(m,8)-T(sh,8)
): LET ty=ABS (T(m,9)-T(sh,9))
1785 IF tm<fx AND T(m,1)<5 AND
ty<fy THEN  LET fx=tm: LET fy=t
y: LET gp=m
1790 NEXT m
1800 IF gp=-1 THEN  PRINT AT 19
,0;"Fora de alcance": GOSUB 241
0: RETURN
1810 LET C=8-T(gp,4)-ABS(fx-fy )
1820 IF gp<3 OR gp=9 OR gp=10 T
HEN  LET C=C+1
1830 IF m(T(gp,8),T(gp,9))=2 TH
EN  LET C=C-2
1840 IF T(gp,1)<>2 THEN  LET C=
C+1
1850 LET C=(C+(INT (T(sh,7)/40)
)+FN r(3))*10
1860 LET T(gp,7)=T(gp,7)-C
1870 PRINT "Houve ";C;" baixas
na unidade ";gp
1875 GOSUB 2410
1880 LET un=gp: GOSUB 2200
1890 RETURN
```

```
1710 REM TIRO
1720 GOSUB 2540
1730 LOCATE 0,19:PRINT "UNIDADE
";SH;"ATIRA"
1740 FX=5:FY=5:GP=-1
1745 ST=9
1750 IF SH>8 THEN ST=1
1770 FOR M=ST TO ST+7
1780 TM=ABS(T(M,8)-(T(SH,8)):TY
=ABS(T(M,9)-T(SH,9))
1785 IF TM<FX AND T(M,1)<5 AND
TY<FY THEN FX=TM:FY=TY:GP=M
1790 NEXT M
1800 IF GP=-1 THEN LOCATE 0,20:
PRINT "FORA DE ALCANCE":GOSUB 2
410
1810 C=8-T(GP,4)-ÀBS(FX-FY)
1820 IF GP<3 OR GP=9 OR GP=10 T
HEN C=C+1
1830 IF M(T(GP,8),T(GP,9))=2 TH
EN C=C-2
1840 IF T(GP,1)<>2 THEN C=C+1
1850 C=(C+INT(T(SH,7)/40))+FN R
(3))*10
1860 T(GP,7)=T(GP,7)-C
1870 PRINT "HOUVE";C;"BAIXAS NA
 NIDADE";GP
1875 GOSUB 2410
1880 UN=GP:GOSUB 2200
1890 RETURN
```




```
1710  REM  TIRO
1720  GOSUB 2540
1730  VTAB 21: PRINT "UNIDADE "
;SH;" ATIRA": GOSUB 30000
1740 FX = 5:FY = 5:GP =  - 1
1745 ST = 9
1750  IF SH > 8 THEN ST = 1
1770  FOR M = ST TO (ST + 7)
1780 TM =  ABS (T(M,8) - T(SH,8
)):TY =  ABS (T(M,9))
1785  IF TM < FX AND T(M,1) < 5
 AND TY < FY THEN FX = TM:FY =
TY:GP = M
1790  NEXT M
1800  IF GP =  - 1 THEN  PRINT
```

```
"FORA DE ALCANCE": GOSUB 30000:
 GOSUB 2410: RETURN
1810 C = 8 - T(GP,4) -  ABS (FX
 - FY)
1820  IF GP < 3 OR GP = 9 OR GP
 = 10 THEN C = C + 1
1830  IF M(T(GP,8),T(GP,9)) = 2
 THEN C = C - 2
1840  IF T(GP,1) <  > 2 THEN C
= C + 1
1850 C = (C + ( INT (T(SH,4) /
40)) +  FN R(3)) * 10
1860 T(GP,7) = T(GP,7) - C
1870  PRINT "HOUVE ";C;" BAIXAS
 NA UNIDADE ";GP
1875  GOSUB 30000: GOSUB 2410
1880 UN = GP: GOSUB 2200
1890  RETURN
```

Modifique as seguintes linhas do programa destinado ao Apple:

```
1730  VTAB 21: PRINT "UNIDADE "
;SH;" ATIRA"
1800  IF GP =  - 1 THEN  PRINT
"FORA DE ALCANCE": GOSUB 2410:
RETURN
1875  GOSUB 2410
```

```
1710 REM TIRO
1720 GOSUB 2540
1730 DRAW"BM0,152":A$="UNIDADE"
+STR$(SH)+" ATIRA":GOSUB 3190
1740 FX=5:FY=5:GP=-1
1745 ST=9
1750 IF SH>8 THEN ST=1
1770 FOR M=ST TO (ST+7)
1780 TM=ABS(T(M,8)-T(SH,8)):TY=
ABS(T(M,9)-T(SH,9))
1785 IF TM<FX AND T(M,1)<5 AND
TY<FY THEN FX=TM:FY=TY:GP=M
1790 NEXT M
1800 IF GP=-1 THEN DRAW"BM0,160
":A$="FORA DE ALCANCE":GOSUB 31
90:GOSUB 2410:RETURN
1810 C=8-T(GP,4)-ABS(FX-FY)
1820 IF GP<3 OR GP=9 OR GP=10 T
HEN C=C+1
1830 IF M(T(GP,8),T(GP,9))=2 TH
EN C=C-2
1840 IF T(GP,1)<>2 THEN C=C+1
1850 C=(C+INT(T(SH,7)/40))+RND(
3))*10
1860 T(GP,7)=T(GP,7)-C
1870 DRAW"BM0,160":A$="HOUVE"+S
TR$(C)+" BAIXAS NA UNIDADE"+STR
$(GP):GOSUB 3190
1875 GOSUB 2410
1880 UN=GP:GOSUB 2200
1890 RETURN
```

A rotina de tiro é chamada sempre que os arqueiros recebem a ordem "FOGO". Em outros jogos pode haver mais de um tipo de unidade com capacidade de atirar diferentes projéteis.

Quando uma unidade recebe a ordem de disparo, **G% (GP** ou **gp)** passa a valer −1 na linha 1740. A seguir, a rotina verificará se há um alvo vulnerável na área. As coordenadas da unidade que atira são comparadas às de cada unidade inimiga pela linha 1780. Se o alvo estiver dentro de um alcance de cinco posições do mapa, **G% (GP** ou **gp)** assume o número da unidade atingida. Se houver mais de um alvo, o mais próximo será considerado.

Caso não haja unidades inimigas por perto, **G% (GP** ou **gp)** permanece valendo −1 e o jogador é informado que o inimigo está "FORA DE ALCANCE".

Se o disparo atingir o alvo, as baixas inimigas serão calculadas e armazenadas em **C%** (ou **C**).

Vários fatores determinam a gravidade dos danos causados por um disparo. Na linha 1810, o tipo de armadura da unidade-alvo é subtraído de 8. Em seguida, subtrai-se do resultado um fator de distância. A linha 1820 adiciona 1 a **C%** (ou **C**), se a unidade-alvo for de cavaleiros. Se ela estava resguardada por uma floresta (terreno tipo 2), a linha 1830 subtrai 2; se se encontrava em movimento, a linha 1840 adiciona 1 (tropas em movimento são mais vulneráveis à artilharia).

Finalmente, a linha 1850 soma o resultado a um quarto do poder da unidade atacante, mais uma parcela aleatória — tudo isso vezes dez. O resultado corresponde ao total de baixas da unidade-alvo. As baixas são subtraídas do poder da unidade e, em seguida, a rotina que testa o moral é chamada.

O CONFRONTO

O resultado do combate corpo a corpo é calculado de maneira similar:

```
1510 REM  Combate
1520 IF (us<9 AND th<9) OR (us>
8 AND th>8) THEN  RETURN
1530 IF T(us,1)=5 OR T(th,1)=5
THEN  RETURN
1540 GOSUB 2540
1550 PRINT AT 18,0;"Combate !!"
1560 LET at=INT ((T(us,7)-T(th,
7))/50)
1570 LET at=at+T(us,3)-T(th,4)+
T(us,5)+FN r(5)
1580 IF ABS (T(us,2)-T(th,2))<>
2 THEN  LET at=at+2
1590 IF us<3 OR us=9 OR us=10 T
HEN  LET at=at+1
1600 LET dr=INT ((T(th,7)-T(us,
7))/60)
1610 LET df=df+T(th,3)-T(us,4)+
T(th,5)+m(T(th,8),T(th,9))+FN r
(3)+2
1615 LET wn=th: LET lo=us
1620 IF at>df THEN  LET wn=us:
LET lo=th
1630 LET wc=INT (T(wn,7)/10): I
F wc<1 THEN  LET wc=1
1640 LET T(wn,7)=T(wn,7)-wc
1650 LET lc=INT (T(lo,7)/5): IF
 lc<1 THEN  LET lc=1
```

```
1660 LET T(lo,7)=T(lo,7)-lc
1670 PRINT wn;" perde ";wc;" ";
lo;" perde ";lc
1680 GOSUB 2410
1690 LET un=lo: GOSUB 2200
1700 RETURN
```

```
1510 REM COMBATE
1520 IF (US<9 AND TH<9) OR (US>
8 AND TH>8) THEN RETURN
1530 IF T(US,1)=5 OR T(TH,1)=5
THEN RETURN
1540 GOSUB 2540
1550 LOCATE 0,19:PRINT "COMBATE
 !!!"
1560 AT=INT((T(US,7)-T(TH,7))/5
0)
1570 AT=AT+T(US,3)-T(TH,4)+T(US
,5)+FN R(5)
1580 IF ABS(T(US,2)-T(TH,2))<>2
 THEN AT=AT+2
1590 IF US<3 OR US=9 OR US=10 T
HEN AT=AT+1
1600 DR=INT((T(TH,7)-T(US,7))/6
0)
1610 DF=DF+T(TH,3)-T(US,4)+T(TH
,5)+M(T(TH,8),T(TH,9))+FN R(3)+
2
1615 WN=TH:LO=US
1620 IF AT>DF THEN WN=US:LO=TH
1630 WC=INT(T(WN,7)/10):IF WC<1
 THEN WC=1
1640 T(WN,7)=T(WN,7)-WC
1650 LC=INT(T(LO,7)/5):IF LC<1
THEN LC=1
1660 T(LO,7)=T(LO,7)-LC
```

```
1670 PRINT WN;"PERDE";WC;".";LO
;"PERDE";LC
1680 GOSUB 2410
1690 UN=LO:GOSUB 2200
1700 RETURN
```

```
1510  REM  COMBATE
1520  IF (US < 9 AND TH < 9) OR
 (US > 8 AND TH > 8) THEN  RETU
RN
1530  IF T(US,1) = 5 OR T(TH,1)
 = 5 THEN  RETURN
1540  GOSUB 2540
1550  VTAB 21: PRINT "COMBATE !
!!": GOSUB 30000
1560 TT =  INT ((T(US,7) - T(TH
,7)) / 50)
1570 TT = TT + T(US,3) - T(TH,4
) + T(US,5) +  FN R(5)
1580  IF  ABS (T(US,2) - T(TH,2
)) <  > 2 THEN TT = TT + 2
1590  IF US < 3 OR US = 9 OR US
 = 10 THEN TT = TT + 1
1600 DR =  INT ((T(TH,7) - T(US
,7)) / 60)
1610 DF = DF + T(TH,3) - T(US,4
) + T(TH,5) + M(T(TH,8),T(TH,9)
) +  FN R(3) + 2
1615 WN = TH:LO = US
1620  IF TT > DF THEN WN = US:L
O = TH
1630 WC =  INT (T(WN,7) / 10):
IF WC < 1 THEN WC = 1
1640 T(WN,7) = T(WN,7) - WC
1650 LC =  INT (T(LO,7) / 5): I
F LC < 1 THEN LC = 1
```

```
1660 T(LO,7) = T(LO,7) - LC
1670  PRINT WN;" PERDE ";WC;" "
;LO;" PERDE ";LC: GOSUB 30000
1680  GOSUB 2410
1690 UN = LO: GOSUB 2200
1700  RETURN
```

Substitua as seguintes linhas:

```
1550  VTAB 21: PRINT "COMBATE !
!!"
1670  PRINT WN;" PERDE ";WC;" "
;LO;" PERDE ";LC
```

```
1510 REM  COMBATE
1520 IF (US<9 AND TH<9) OR (US>
8 AND TH>8) THEN RETURN
1530 IF T(US,1)=5 OR T(TH,1)=5
THEN RETURN
1540 GOSUB 2540
1550 DRAW"BM0,144":A$="COMBATE!
!":GOSUB 3190
1560 AT=INT((T(US,7)-T(TH,7))/5
0)
1570 AT=AT+T(US,3)-T(TH,4)+T(US
,5)+RND(5)
1580 IF ABS(T(US,2)-T(TH,2))<>2
 THEN AT=AT+2
1590 IF US<3 OR US=9 OR US=10 T
HEN AT=AT+1
1600 DF=INT((T(TH,7)-T(US,7))/6
0)
1610 DF=DF+T(TH,3)-T(US,4)+T(TH
,5)+M(T(TH,8),T(TH,9))+RND(3)+2
```

```
1615 WN=TH:LO=US
1620 IF AT>DF THEN WN=US:LO=TH
1630 WC=INT(T(WN,7)/10):IF WC<1
 THEN WC=1
1640 T(WN,7)=T(WN,7)-WC
1650 LC=INT(T(LO,7)/5):IF LC<1
THEN LC=1
1660 T(LO,7)=T(LO,7)-LC
1670 DRAW"BM0,160":A$=STR$(WN)+
" PERDE"+STR$(WC)+" "+STR$(LO)+
" PERDE"+STR$(LC):GOSUB 3190
1680 GOSUB 2410
1690 UN=LO:GOSUB 2200
1700 RETURN
```

Essa rotina é chamada sempre que duas unidades se encontram no tabuleiro. Se elas forem do mesmo exército, a rotina é imediatamente interrompida. O mesmo acontece se uma das unidades estiver em retirada — linha 1530.

A principal diferença entre o combate balístico e o corpo a corpo é que, no segundo caso, as duas unidades entram em ação. O combate corpo a corpo é, portanto, mais complexo, exigindo que se calculem as baixas dos dois lados.

A linha 1560 dá ao atacante uma nota igual a um quinto da diferença entre o poder das duas unidades. Em seguida, a diferença entre a arma do atacante e a armadura do atacado é somada ao valor do moral do atacante mais um número aleatório de 1 a 5.

Aquele que ataca tem um bônus adicional se o inimigo não estiver de frente. Inclui-se nessa situação o atacante que se movia em direção diferente da do inimigo — se ele estiver parado, a direção de seu último movimento será considerada. Por isso, nunca apagamos o elemento de direção de uma unidade, após uma ordem "ALTO". Finalmente, o atacante tem também um bônus se for uma unidade de cavaleiros.

A nota da unidade atacada é calculada de modo semelhante, correspondendo à soma destes elementos: um sexto da diferença de poder, a diferença entre a arma do atacante e a armadura do atacado, o moral do defensor, um bônus que depende do terreno, um bônus fixo de 2, um número aleatório de 1 a 3. Esses cálculos têm como pressuposto que defender é mais fácil que atacar. Às vezes, porém, o entusiasmo do atacante pode neutralizar tal vantagem.

Assim, os atacantes obterão uma nota (**AT**) e os defensores outra (**DF**). A unidade que alcança o valor mais alto ganha a batalha, perdendo apenas um décimo de seu poder. Já o poder do derrotado é reduzido em um quinto.

Finalmente, a rotina que testa o moral é chamada para o derrotado.

O MORAL

O fator psicológico tem grande peso numa guerra. É bem possível que um exército seja derrotado, mesmo que esteja equipado com as melhores armas do mundo, se a tropa detestar seus generais, simpatizar com a causa inimiga ou, simplesmente, não quiser lutar.

A psicologia humana é muito complexa e não pretendemos reproduzi-la aqui. Em *Capa e Espada* o moral influencia apenas a sobrevivência da unidade e o resultado do combate.

```
2200 REM   Moral
2210 IF T(un,6)-T(un,7)<((T(un,
6)/100)*((T(un,5)+2))) THEN  RE
TURN
2220 GOSUB 2540
2230 PRINT AT 18,0;" As perdas
foram  tao  grandes"
2240 PRINT AT 19,0;" que a unid
ade ";un;" se desintegra"
2250 GOSUB 2410
2260 LET T(un,1)=5
2270 PRINT AT T(un,8),T(un,9);"
 "
2280 RETURN
```

```
2200 REM MORAL
2210 IF T(UN,6)-T(UN,7)<((T(UN,
6)/100)*((T(UN,5)+2))) THEN RET
URN
2220 GOSUB 2200
2230 LOCATE 0,19:PRINT "AS PERD
AS FORAM TAO PESADAS"
```

```
2240 PRINT "QUE A UNIDADE";UN;"
SE DESINTEGRA"
2250 GOSUB 2410
2260 T(UN,1)=5
2270 LOCATE T(UN,9),T(UN,8):PRI
NT " ";
2280 RETURN
```

```
2200  REM  MORAL
2210  IF T(UN,6) - T(UN,7) < ((
T(UN,6) / 100) * ((T(UN,5) + 2)
)) THEN  RETURN
2220  GOSUB 2540
2230  VTAB 21: PRINT "AS BAIXAS
 FORAM MUITO GRANDES"
2240  PRINT "A UNIDADE ";UN;" S
E DESINTEGROU"
2250  GOSUB 2410
2260 T(UN,1) = 5
2270 X = T(UN,9):Y = T(UN,8):N
= 14: GOSUB 10000
2280  RETURN
```

Adicione as seguintes linhas:

```
2545  COLOR= 0: FOR I = 160 TO
191
2550  HPLOT 0,I TO 279,I: NEXT
: RETURN
30000  RETURN
```

```
2200 REM  MORAL
2210 IF T(UN,6)-T(UN,7)<((T(UN,
6)/100)*((T(UN,5)+2)*10)) THEN
RETURN
2220 GOSUB 2540
2230 DRAW"BM0,152":A$="AS PERDA
S FORAM MUITO GRANDES":GOSUB 31
90
2240 DRAW"BM0,160":A$="UNIDADE"
+STR$(UN)+" SE DESINTEGRA":GOSU
B 3190
2250 GOSUB 2410
2260 T(UN,1)=5
2270 X9=T(UN,9)*8:Y9=T(UN,8)*8:
LINE(X9,Y9)-(X9+7,Y9+7),PRESET,
BF
2280 RETURN
```

A linha 2210 subtrai o poder atual do poder inicial da tropa, comparando o resultado com o moral. Se a unidade sofreu uma perda de 30% e tem o moral baixo, ela se dispersa e não participa mais do jogo. Se o moral for mais elevado, uma unidade poderá perder até 70% do poder antes de se dispersar.

Quando uma unidade não passa no teste do moral, as linhas 2230 e 2240 transmitem uma mensagem informando a dispersão. O elemento de comando da matriz da tropa passa a valer 5, o que significa que a tropa bateu em retirada. A unidade é apagada da tela e ignorada. Porém, como há desertores espalhados pelo campo de batalha, a unidade poderá dificultar o movimento de tropas, apesar de estar invisível.

ENFIM, A PAZ

Entre os últimos detalhes que precisamos incluir no programa está uma rotina de verificação do fim do jogo.

```
2290 REM  Vitoria
2300 LET gd=0: LET bd=0
2310 FOR m=1 TO 8
2320 IF T(m,1)<>5 THEN  LET gd=
gd+1
2330 IF T(m+8,1)<>5 THEN  LET b
d=bd+1
2340 NEXT m
2350 IF gd>bd*2 OR (bd<2 AND gd
>2) THEN  LET vc=1
2360 IF bd>gd*2 OR (gd<2 AND bd
>2) THEN  LET de=1
2370 RETURN
2380 REM  Fim
2390 IF vc=1 THEN  PRINT "VITOR
IA !!!"
2395 IF de=1 THEN  PRINT "Uma h
umilhante derrota."
2400 RETURN
```

```
2290 REM VITÓRIA
2300 GD=0:BD=0
2310 FOR M=1 TO 8
2320 IF T(M,1)<>5 THEN GD=GD+1
2330 IF T(M+8,1)<>5 THEN BD=BD+
1
2340 NEXT M
2350 IF GD>BD*2 OR (BD<2 AND GD
>2) THEN VC=1
2360 IF BD>GD*2 OR (GD<2 AND BD
>2) THEN DE=1
2370 RETURN
2380 REM FIM
2390 IF VC=1 THEN PRINT "VITÓRI
A !!!"
2395 IF DE=1 THEN PRINT "UMA HU
MILHANTE DERROTA."
2400 RETURN
```

```
2290  REM  VITORIA
2300 GD = 0:BD = 0
2310  FOR M = 1 TO 8
2320  IF T(M,1) <  > 5 THEN GD
= GD + 1
2330  IF T(M + 8,1) <  > 5 THEN
 BD = BD + 1
2340  NEXT M
2350  IF GD > BD * 2 OR (BD < 2
 AND GD > 2) THEN VC = 1
2360  IF BD > GD * 2 OR (GD < 2
 AND BD > 2) THEN DE = 1
2370  RETURN
2380  REM  FIM
2390  IF VC = 1 THEN  PRINT "VI
TORIA !!!": GOSUB 30000
2395  IF DE = 1 THEN  PRINT "UM
A HUMILHANTE DERROTA": GOSUB 30
000
2400  RETURN
```

Modificações do programa do Apple:

```
2390  IF VC = 1 THEN  PRINT "VI
TORIA !!!"
2395  IF DE = 1 THEN  PRINT "UM
A HUMILHANTE DERROTA"
```

```
2290 REM  VITORIA
2300 GD=0:BD=0
2310 FOR M=1 TO 8
2320 IF T(M,1)<>5 THEN GD=GD+1
2330 IF T(M+8,1)<>5 THEN BD=BD+
1
2340 NEXT M
2350 IF GD>BD*2 OR (BD<2 AND GD
>2) THEN VC=1
2360 IF BD>GD*2 OR (GD<2 AND BD
>2) THEN DE=1
2370 RETURN
2380 REM  FIM
2390 IF VC=1 THEN DRAW"BM0,176"
:A$="VITORIA !!!":GOSUB 3190
2395 IF DE=1 THEN DRAW"BM0,176"
:A$="UMA HUMILHANTE DERROTA.":G
OSUB 3190
2400 RETURN
```

A rotina verifica se um dos adversários conta com o dobro de unidades do outro ou se um deles tem menos de duas unidades. Uma mensagem informa aos jogadores o resultado.

Poderíamos acrescentar outras condições de vitória. Por exemplo, um fator que decidiu muitas guerras do período medieval foi a morte do líder. Porém, a incorporação desse elemento a um jogo, além de trazer outros problemas, faria com que a atenção dos jogadores estivesse permanentemente voltada para uma das unidades.

Uma unidade também poderia obter a vitória se atingisse a extremidade oposta do mapa com metade de seus efetivos, ou metade de seu poder. Outro critério seria pelo total de baixas. Nesse caso, teríamos de acrescentar uma variável para calcular o número.

ALEA JACTA EST

Já digitamos todas as rotinas necessárias ao jogo. Agora, precisamos apenas incluir um laço principal, que as chame ordenadamente:

```
10 CLEAR
30 GOSUB 190
```

```
  40 GOSUB 470
  50 GOSUB 860
  60 REM  Loop
  70 FOR i=1 TO 8
  80 IF T(i,1)<4 THEN  GOSUB
  1380: IF y>2 THEN  GOSUB 1900
  90 IF T(i,1)<5 THEN  INK 1:
  PRINT AT T(i,8),T(i,9);u$(i)
 100 NEXT i
 110 FOR e=9 TO 16
 120 IF T(e,1)<4 THEN  GOSUB
  2140
 130 NEXT e
 140 GOSUB 1020
 150 GOSUB 2290
 160 IF vc<>1 AND de<>1 THEN
 GOTO 60
 170 GOSUB 2380
 180 STOP
2410 REM  Atraso
2420 PRINT AT 21,7;"(QUALQUER T
ECLA)"
2425 LET q$=INKEY$: IF q$="" TH
EN  GOTO 2425
2430 RETURN
```

```
10 CLEAR 5000
30 GOSUB 190
40 GOSUB 470
50 GOSUB 860
70 FOR I=1 TO 8
80 IF T(I,1)<4 THEN GOSUB 1380:
IF YW>2 THEN GOSUB 1900
90 IF T(I,1)<5 THEN LOCATE T(I,
9),T(I,8):PRINT CHR$(U(I));
100 NEXT I
110 FOR E=9 TO 16
120 IF T(E,1)<4 THEN GOSUB 2140
130 NEXT E
140 GOSUB 1020
150 GOSUB 2290
160 IF VC<>1 AND DE<>1 THEN 60
170 GOSUB 2380
180 END
2410 REM ATRASO
2420 LOCATE 5,22:PRINT "APERTE
QUALQUER TECLA"
2425 G$=INKEY$:IF G$="" THEN 24
25
2430 RETURN
```

```
30  GOSUB 190
40  GOSUB 470
50  GOSUB 860
70  FOR I = 1 TO 8
80  IF T(I,1) < 4 THEN  GOSUB 1
380: IF YW > 2 THEN  GOSUB 1900
90  IF T(I,1) < 5 THEN X = T(I,
9):Y = T(I,8):N =  VAL ( MID$ (
U$,I,1)): GOSUB 10000
100  NEXT I
110  FOR E = 9 TO 16
120  IF T(E,1) < 4 THEN  GOSUB
2140
130  NEXT E
140  GOSUB 1020
150  GOSUB 2290
160  IF VC <  > 1 AND DE <  > 1
 THEN  GOTO 70
170  GOSUB 2380
180  END
475  REM  XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXX
476  REM  XXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXX
2410  REM   ATRASO
2420  PRINT "APERTE QUALQUER TE
CLA";: GOSUB 30000
2425  GET G$:
2430  RETURN
30000  FOR QQ = 1616 TO 2000 ST
EP 128
30010  FOR WW = 0 TO 39
30020  POKE QQ + WW + 1024, PEE
K (QQ + WW)
30030  NEXT WW,QQ
30040  RETURN
```

Modificações a serem feitas no programa acima:

```
2420  PRINT "APERTE QUALQUER TE
CLA";
```

```
10 CLEAR 500:PMODE 3,1:COLOR2,1
:PCLS:SCREEN 1,0:DU=RND(-TIMER)
18 GOSUB 2410:CLS:END
30 GOSUB 190
40 GOSUB 470:GOSUB 3130
50 GOSUB 860
60 REM
70 FOR I=1 TO 8
80 IF T(I,1)<4 THEN GOSUB 1380:
IF Y>2 THEN GOSUB 1900
90 IF T(I,1)<5 THEN COLOR 3:DRA
W"BM"+STR$(T(I,9)*8)+","+STR$(T
(I,8)*8):UU=VAL(MID$(U$,I,1)):A
$=UC$(UU):GOSUB 3000
100 NEXT I
110 FOR E=9 TO 16
120 IF T(E,1)<4 THEN GOSUB 2140
130 NEXT E
140 GOSUB 1020
150 GOSUB 2290
160 IF VC<>1 AND DE<>1 THEN 60
170 GOSUB 2380
190 REM  INICIO
200 VC=0:DE=0
2410 REM  ATRASO
2420 DRAW"BM80,176":A$="APERTE
QUALQUER TECLA":GOSUB 3190
2425 G$=INKEY$:IF G$="" THEN 24
25
2426 LINE(0,176)-(255,183),PRES
ET,BF
2430 RETURN
```

Em primeiro lugar, todos os microcomputadores, com exceção dos pertencentes às linhas Apple e TK-2000, limpam a memória. O TRS-Color seleciona o modo gráfico. Em seguida, as rotinas que criam os blocos gráficos e desenham o mapa são chamadas.

O laço principal começa na linha 60 e termina na linha 160. Ele permite que tanto o jogador quanto o computador dêem ordens às suas respectivas unidades. Depois, a rotina que faz com que as unidades obedeçam às ordens é executada dezesseis vezes, seguida de uma chamada para verificação de final de jogo. O laço se repete enquanto um dos lados não vencer.

Ao final da listagem (começando na linha 2410), temos uma rotina encarregada de provocar uma pausa em determinados momentos do jogo.

AS INSTRUÇÕES

Ao ser executado, o programa desenha o mapa com os diferentes terrenos, a moldura e as tropas beligerantes.

Este é o momento de planejar sua estratégia. Uma série de mensagens surgirá na janela de texto. Começando com a unidade um, são mostrados na tela o número e o tipo de homens da unidade, juntamente com suas últimas ordens — "ALTO", por exemplo. O jogador é então questionado sobre uma possível mudança de ordens.

Se a resposta for Sim, um menu com as opções FOGO, ALTO, MARCHE e STATUS é exibido. A opção de atirar aplica-se somente aos arqueiros e, se selecionada indevidamente para qualquer outra unidade, surgirá na tela a mensagem "SEM ARCOS". Quando a opção MARCHE for selecionada, o computador pedirá que se indique a direção (N,S,L,O).

Esse processo se repete para cada unidade, cuja cor é modificada para orientar o jogador.

O QUE FALTA?

Neste estágio, *Capa e Espada* é um jogo de guerra bem simples, mas, ainda assim, bastante divertido.

Na última parte da série, veremos como transformar o computador em um jogador mais habilidoso. Mas, especialmente se você for um principiante, tente jogar *Capa e Espada* como está, para apreciar os principais aspectos desse tipo de jogo.

AVALANCHE: O TEMPO FECHA

Meteorologia pode não ser seu ponto forte, mas chegou a hora de adicionar um pouco de tempestade ao jogo. Avalanche tem agora nuvens ou sol, conforme a versão do programa.

Como se Willie já não tivesse problemas suficientes com os buracos, as serpentes e a maré, uma nuvem escura promete chuva, ameaçando molhar seu lanche! Mas não se preocupe se você não for usuário do Spectrum ou do MSX. Troveja apenas nas versões de *Avalanche* para esses micros.

Willie está com mais sorte no programa do TRS-Color. Não há nuvens no céu e um belo sol de verão brilhará durante todo o jogo.

A seguinte rotina desenha uma nuvem que se movimenta de acordo com a direção do vento:

```
 10 REM org 58795
 20 REM cld ld a,(57347)
 30 REM dec a
 40 REM ld (57347),a
 50 REM cp 0
 60 REM jr z,cdm
 70 REM ret
 80 REM cdm ld a,6
 90 REM ld (57347),a
100 REM ld a,45
110 REM ld bc,16384
120 REM ld hl,(57345)
130 REM ld d,3
140 REM ld e,2
150 REM call blk
160 REM ld a,(57348)
170 REM cp 0
180 REM jr z,crt
190 REM dec hl
200 REM jr chm
210 REM crt inc hl
220 REM chm ld (57345),hl
230 REM ld bc,57144
240 REM ld a,47
250 REM ld d,3
260 REM ld e,2
270 REM call blk
280 REM ld de,129
290 REM sbc hl,de
300 REM jr nz,cnr
310 REM ld a,0
320 REM ld (57348),a
330 REM ret
340 REM cnr ld de,144
350 REM ld hl,(57345)
360 REM sbc hl,de
370 REM jr nz,cnl
380 REM ld a,1
390 REM ld (57348),a
400 REM cnl ret
410 REM org 58970
420 REM blk *
```

A posição de memória 57347 contém o atraso da nuvem. Essa variável controla a rapidez com que ela se movimenta pelo céu. Quando se inicializa o nível do jogo, a velocidade é especificada por meio de um número que é carregado nessa posição.

Depois de carregado no acumulador, o atraso da nuvem é decrementado e armazenado de volta em 57347. Se seu valor tiver sido reduzido a zero, as instruções **cp 0** e **jr z** saltam a instrução **ret** e dão prosseguimento à rotina que movimenta a nuvem. Caso contrário, o processador retorna e adia o movimento da nuvem até que o valor do atraso tenha sido reduzido a zero.

CÉU AZUL

A rotina encarregada de movimentar a nuvem começa acertando a variável de atraso. Repare que, se o valor desta já foi reduzido a 0, um outro decremento resultaria em 255. Para que o atraso chegasse novamente a 0, a rotina precisaria ser chamada 256 vezes — o que tornaria o movimento da nuvem extremamente lento. Para evitar que isso ocorra, o número 6 é carregado no acumulador e colocado no endereço 57347. Com o atraso assim restabelecido, uma suave e refrescante brisa passa a empurrar a nuvem, o que você pode constatar observando a tela.

O número 45 é carregado no acumulador. Ele será aproveitado pela rotina **blk**, que imprime um bloco de caracteres. As dimensões do bloco devem ser carregadas nos registradores D e E — D carrega o número de colunas e E, o número de linhas. A rotina **blk**, por sua vez, utiliza a rotina **print**, que tem sido empregada com freqüência neste jogo. O par de registros HL deve, então, carregar a posição de tela; o par BC, o apontador de dados, e o acumulador A, a cor do caractere.

O número 5 — código da cor azul ciano sobre azul ciano — é carregado no acumulador, pois a rotina imprime caracteres de céu sobre a nuvem, apagando-a. Depois disso, imprime a nuvem uma posição para a direita ou para a esquerda, dependendo da direção em que o vento está soprando.

O par de registros BC é carregado com o endereço inicial dos padrões. Esse endereço irá fornecer os dados apropriados ao bloco azul de céu. O par HL é carregado com o conteúdo de 57345 e 57346, que são os apontadores da posição atual da nuvem.

A nuvem tem três colunas de comprimento por duas linhas de altura. Assim, 3 é carregado em D e 2, em E. Em seguida, a rotina **blk** é chamada e imprime o bloco de caracteres de céu azul, apagando a nuvem.

DIREÇÃO DO VENTO

A nuvem deve se movimentar na direção em que o vento está soprando. Para isso, utiliza-se a posição de memória 57348, originalmente ajustada pela rotina de inicialização, como uma baliza. Se ela contém o valor 1, o vento está soprando para a esquerda; se contém o valor 0, o vento está soprando para a direita.

O conteúdo de 57348 é carregado no acumulador. Se seu valor for 0, a instrução **jr z** salta para o rótulo **crt** e o apontador de posição da nuvem é incrementado, movendo-se uma posição para a direita na tela. Se o conteúdo de 57348 for 1, a instrução **jr z** não atua e o par de registros HL é decrementado, movendo o apontador de posição da nuvem um caractere para a esquerda. A próxima instrução **jr z** simplesmente salta sobre a instrução **inc**.

SOPRANDO AS NUVENS

O conteúdo de HL é trazido de volta para o apontador de posição da nuvem, nos endereços 57345, ajustando-o. Em seguida, o apontador de dados BC é carregado com o endereço inicial dos dados da nuvem.

O acumulador é carregado com 47 — código de branco sobre fundo ciano, a cor da nuvem. O registro D é carregado com 3 e o registro E, novamente com 2.

- DESENHO DA NUVEM
- DIREÇÃO DO VENTO
- OS CARACTERES DO CÉU AZUL
- O EFEITO DO ATRASO
- MUDANÇA DE DIREÇÃO
- CÉU ENCOBERTO
- O MOVIMENTO DAS NUVENS
- VERIFICAÇÃO DA POSIÇÃO
- O SOL NO TRS-COLOR

A nuvem é do mesmo tamanho daquela que foi apagada.

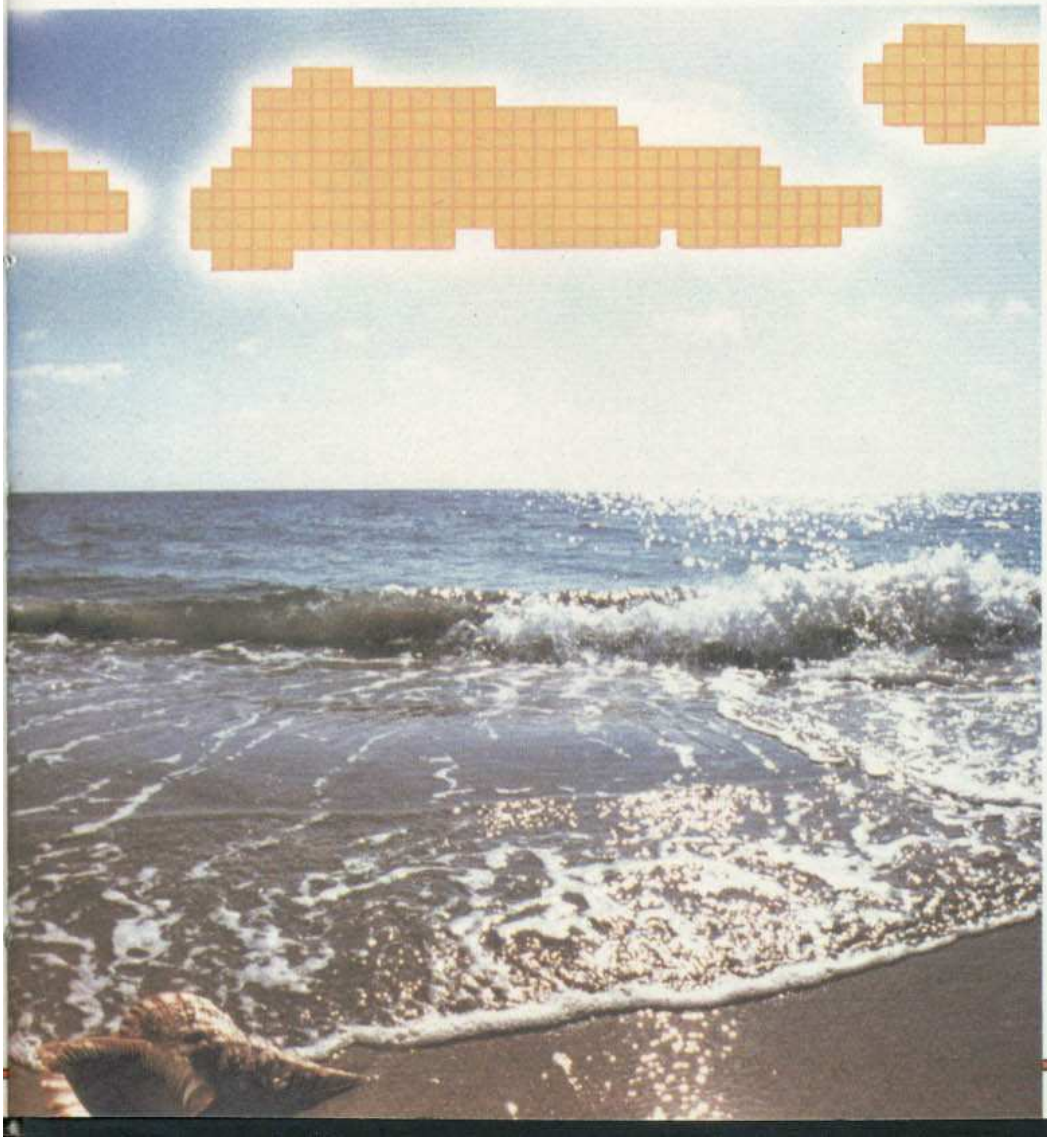
Depois disso, a rotina **blk** é chamada e imprime a nuvem em sua nova posição na tela.

O movimento da nuvem pode levá-la até o canto do vídeo. A parte dela que ultrapassar a primeira ou a última coluna será impressa na linha seguinte ou na linha de cima, provocando a desfiguração de sua forma.

Para evitar que isso aconteça, o programa verifica primeiro se a nuvem alcançou o extremo esquerdo do vídeo. O par DE é carregado com 129 — a posição de tela do canto esquerdo da linha. Esse valor é subtraído do apontador de posição da nuvem no par HL. Se o resultado não for 0, a nuvem não chegou no canto esquerdo. A instrução **jr nz** salta então para verificar se ela alcançou o canto direito.

Se o resultado for 0, a nuvem atingiu o canto esquerdo e o salto não ocorre. O acumulador é então carregado com 0 e esse valor é transferido para a baliza de direção do vento, de modo que a nuvem se desloque para a direita na próxima vez.

A rotina **crr** verifica se a nuvem chegou ao canto esquerdo do vídeo subtraindo o número 144 do apontador de posição. Se ela atingiu o canto direito da linha, o valor 1 é carregado no indicador da direção do vento. Na próxima vez, a nuvem irá para a esquerda.



O BLOCO DA NUVEM

Esta rotina imprime um bloco de caracteres na tela, com D colunas de comprimento por E linhas de altura.

```
 10 REM org 58970
 20 REM blk push hl
 30 REM blj push de
 40 REM push hl
 50 REM z push de
 60 REM call print
 70 REM inc hl
 80 REM pop de
 90 REM dec d
100 REM jr nz,z
110 REM pop hl
120 REM ld de,32
130 REM add hl,de
140 REM pop de
150 REM dec e
160 REM jr nz,blj
170 REM pop hl
180 REM ret
190 REM org 58217
200 REM print *
```

O programa anterior, que movimenta a nuvem, utiliza a rotina **blk**. Portanto, não irá trabalhar até que a rotina que acabamos de apresentar esteja guardada na memória. Esta, por sua vez, não entrará em funcionamento sem que a rotina **print**, que também é chamada, se encontre na memória.

É O BLOCO CERTO?

O apontador da posição de tela em HL e o número de colunas e linhas que estão em DE são armazenados na pilha duas vezes. Isso é feito porque a rotina trabalhará com duas dimensões, usando dois laços.

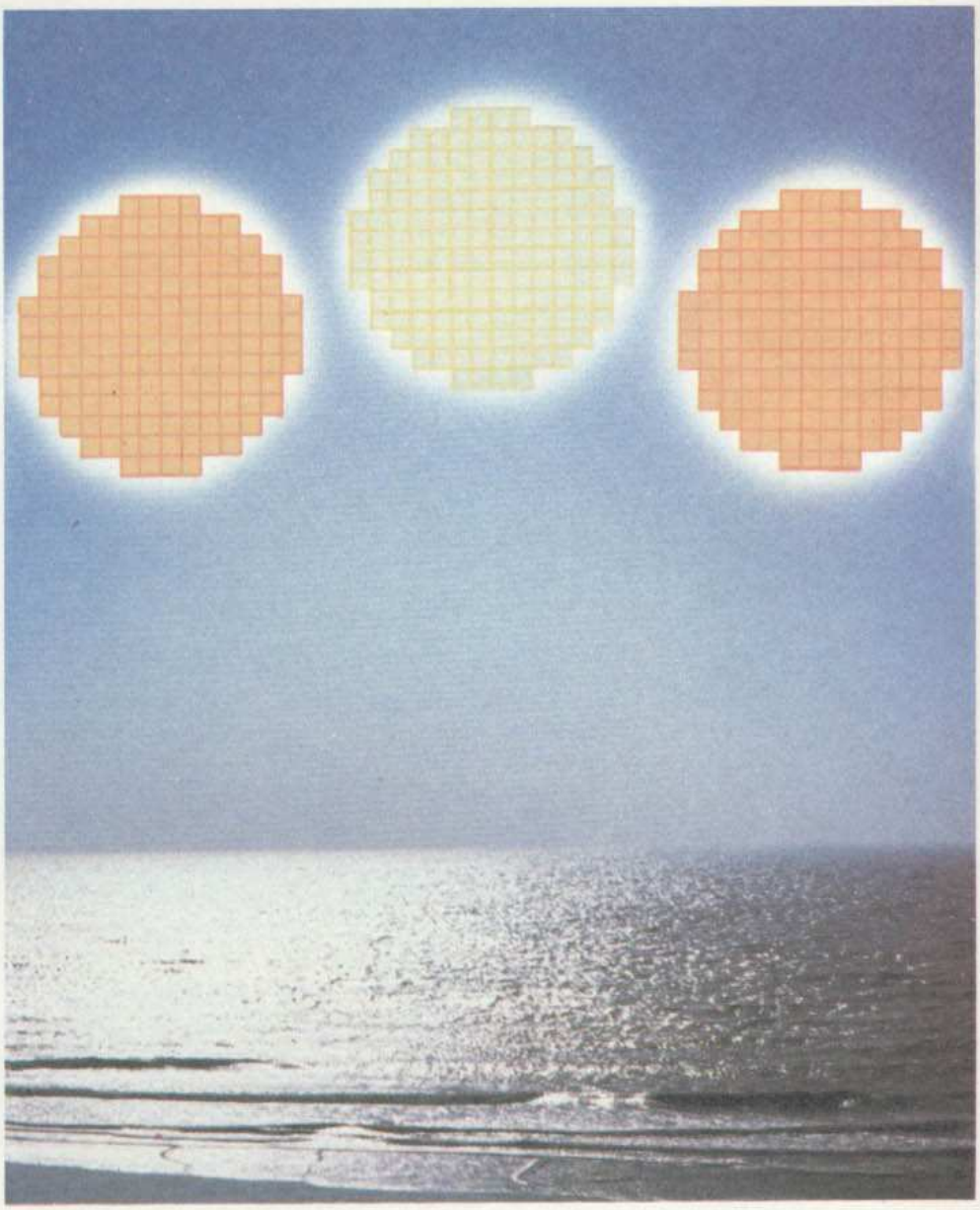
Como os parâmetros já estão armazenados, a rotina **print** é chamada e imprime o primeiro caractere na tela. O par HL é incrementado, movendo-se um caractere para a direita. As dimensões do bloco são recuperadas da pilha e o parâmetro horizontal — equivalente ao número de colunas — é decrementado. Se o parâmetro não foi reduzido a 0, a instrução **jr nz** volta ao início do laço para imprimir outro caractere da nuvem nessa linha da tela.

Quando o registrador D for 0, a primeira linha do bloco estará completa. Em seguida, recupera-se o par HL da pilha e adiciona-se o número 32 ao seu valor. Como se trata de uma adição de números de dezesseis bits, a operação é feita por intermédio do par DE. O resultado fica no par HL, fazendo com que esse apontador de tela se mova uma linha para baixo. O par DE é novamente recuperado da pilha, para ajustar o valor em D e para decrementar o valor em E. Esse registrador conta o número de linhas que falta.

Se o valor no registro E não tiver sido reduzido a 0, a instrução **jr nz** manda o processador de volta ao início do laço, para começar a impressão de uma nova linha.

Quando o valor de E for 0, o processador sai do laço e o par HL é recuperado da pilha, ajustando o apontador de tela para as checagens que serão realizadas. Lembre-se de que precisamos verificar se a nuvem não atingiu nenhum dos cantos da tela.

Depois disso, a rotina **blk** retorna ao programa principal deste artigo.



Na versão de *Avalanche* para o TRS-Color, a inclusão de uma nuvem exigiria um conjunto adicional de dados ou padrões. Ainda que isso não constituísse um problema, haveria um outro obstáculo: o conjunto de cores já definido é azul, vermelho e verde — nenhuma delas, portanto, é adequada para a nuvem. Como nesta versão a aventura transcorre num dia muito quente — com o céu avermelhado —, uma nuvem não é realmente essencial. A rotina a seguir movimenta o sol durante o jogo.

```
10   ORG 19727
20   MOVSUN DEC 18258
30   BNE SUNRET
40   LDA #5
50   STA 18258
60   SYNC
70   LDX #1569
80   LDA #30
90   MOVA PSHS A
100   LDA #2
110   MOVB ANDCC #$FE
120   PSHS CC
130   CLRB
140   MOVC PULS CC
150   ROR B,X
160   PSHS CC
170   INCB
180   CMPB #14
190   BNE MOVC
200   LSL,X
210   PULS CC
220   ROR,X
230   DECA
240   BNE MOVB
250   LEAX 32,X
260   PULS A
270   DEC A
280   BNE MOVA
290   SUNRET RTS
```

Para testar a rotina, digite o seguinte programa:

```
10 EXEC 19426
20 EXEC 19727
30 GOTO 20
```

A posição de memória 18258 contém a variável de atraso do sol. Essa variável impede que o atraso risque o céu como se fosse um disco voador, e é ajustada, inicialmente, com 5.

A primeira instrução da rotina decrementa o atraso do sol. Se o valor dessa variável for 0, a instrução **BNE** é desviada para a instrução **RTS** do final da rotina e o processador retorna. Caso contrário, a instrução **BNE** não realiza o desvio e o processador continua com o restante da rotina. Em outras palavras, o movimento do sol sofre um atraso, já que a rotina é executada uma vez a cada cinco chamadas.

Logo que é chamada, a rotina ajusta o atraso do sol, colocando o número 5 no acumulador e armazenando-o de volta na posição de memória 18258.

SINCRONISMO

O comando **SYNC** encarrega-se de sincronizar o restante da rotina com o sinal de TV. Isso é feito para resguardar a posição do sol da mudança que se efetuará na tela. O registro X é carregado com o endereço na tela do canto superior esquerdo do sol. O acumulador

A é carregado com 30 (o sol tem 30 linhas de altura), valor que se guarda na pilha. Em seguida, A é carregado com 2 — um contador de laço que fará o sol se deslocar lateralmente dois pontos na tela.

A operação AND é efetuada entre o registro de código condicional e o número hexadecimal $FE. Como a representação binária de $FE é 11111110, os sete bits mais significativos do registro de código condicional permanecem inalterados, enquanto o bit menos significativo é ajustado com 0. Esse bit — ou seja, o bit 0 do registro de código condicional — corresponde à baliza **carry**, que, portanto, é zerada nessa operação.

O registro de código condicional é guardado na pilha. O registrador B, ajustado com 0, será usado como um compensador. A rotina está ajustada e pronta para realizar o deslocamento.

ROTAÇÃO DO SOL

A parte seguinte da rotina que estamos examinando desloca o sol pelo céu. Para isso recupera da pilha o registro de código condicional, garantindo que, na primeira passagem do laço, o valor da baliza **carry** seja zero.

A instrução **ROR B,X** roda uma posição para a direita os bits do endereço da tela apontado por X+B. A rotação desloca todos os bits. Assim, o último lugar desocupado no processo é carregado com o conteúdo da baliza **carry**, e o bit que ultrapassou o fim do outro lado é colocado nessa baliza.

Os pontos que formam essa parte do sol são deslocados uma posição para a direita. O ponto no canto esquerdo é aceso (ajustado para 1) ou apagado (ajustado para 0, ou seja, adquire a cor do céu). O resto desse byte do sol é deslocado um ponto.

O ponto colocado na baliza **carry** é preservado e a baliza volta para a pilha. O contador de laço em B é incrementado e comparado a 14 — número de caracteres da parte do céu ocupada pelo sol. A comparação permite-nos verificar se o último byte da área já foi deslocado. Se B for menor que 14, a instrução **BNE** manda o processador de volta, para continuar com o próximo caractere. Chegando ao último caractere, o processador salta do laço.

FINAL LIVRE

Esta aventura se passa no hemisfério norte e o sol se move da esquerda para a direita. Seria muito artificial se, ao alcançar o final da tela ou esbarrar no escore, o sol começasse a fazer o caminho de volta — isto é, da direita para a esquerda.

Porém, como não existe noite na região de *Avalanche*, nosso astro reaparecerá à esquerda sempre que concluir seu percurso na tela.

A instrução **LSL ,X** promove o deslocamento lógico para a esquerda do conteúdo da posição de tela apontada por X. O registrador X não foi alterado nesta parte da rotina, e, por isso, ainda aponta para o canto inferior esquerdo da figura do sol. Assim, o bit do canto esquerdo é deslocado para fora do registrador. O bit do canto direito, por sua vez, que já tinha sido deslocado do byte, é recuperado da pilha, voltando para a baliza **carry**. Em seguida, a instrução **ROR ,X** faz com que ele execute uma rotação sobre o canto esquerdo da posição de tela.

O contador em A é decrementado e, se seu valor ainda não tiver chegado a 0, o processador salta para deslocar o sol mais um ponto.

Quando o deslocamento corresponder a dois pontos, a instrução **LEAX 32,X** adiciona o valor 32 ao conteúdo do registro X e movimenta o apontador da posição de tela para a próxima linha abaixo do sol. O contador de linha, recuperado da pilha para o acumulador, é, então, decrementado. Se seu conteúdo não for igual a 0, a instrução **BNE** manda o processador de volta para rolar a próxima coluna de caracteres. Caso contrário, o processador salta para a instrução **RTS** e retorna.

A rotina a seguir coloca na tela uma nuvem que se move pelo céu, empurrada pelo vento:

```
10    org 54270
20    ld a,(-5208)
30    dec a
40    ld (-5208),a
50    cp 0
60    jr z,cl
70    ret
80    cl ld a,6
90    ld (-5208),a
100    ld a,(-5207)
110    cp 0
120    jr z,ct
130    ld hl,-5210
140    dec (hl)
150    jr cm
160    ct ld hl,-5210
170    inc (hl)
180    cm ld hl,(62413)
190    ld b,6
200    ld a,(-5210)
210    ld c,a
220    ld d,20
230    ld e,1
240    call -11168
250    ld hl,(62413)
260    ld de,4
270    add hl,de
280    ld b,6
290    ld a,(-5210)
300    add a,16
310    ld c,a
320    ld d,24
330    ld e,1
340    call -11168
350    ld a,(-5210)
360    cp 2
370    jr nz,cr
380    ld a,0
390    ld (-5207),a
400    ret
410    cr ld a,(-5210)
420    cp 230
430    jr nz,cf
440    ld a,1
450    ld (-5207),a
460    cf ret
470    end
```

O endereço −5208 contém a variável de atraso da nuvem. Essa variável controla a rapidez com que a nuvem se move. A velocidade foi especificada na parte da rotina principal que inicializa as variáveis, com a armazenagem do número 6 naquele endereço.

O atraso da nuvem é carregado no acumulador, decrementado e novamente armazenado em −5208. Quando a variável é reduzida a 0, as instruções **cp 0** e **jr z** saltam a instrução **ret**, e o processador executa o restante da rotina que movimenta a nuvem. Enquanto isso não ocorre, o processador retorna e adia o movimento da nuvem. Em outras palavras, a rotina é executada uma vez a cada seis chamadas. Utilizamos essa técnica de controle do sincronismo em várias partes deste jogo.

DIREÇÃO DO VENTO

Antes de mais nada, a rotina acerta o atraso da nuvem. Lembre-se de que, para que ela prossiga, o conteúdo da variável de atraso precisa ser reduzido a zero. Um outro decremento teria como resultado 255, pois, na representação binária adotada pelo processador, −1 é o mesmo que 255. Assim, para uma nova execução, a rotina teria que ser chamada 255 vezes, o que resultaria num movimento extremamente lento da nuvem. Por esse motivo, o número 6 é carregado em −5208 pelo acumulador.

Em seguida, o conteúdo de −5207 é colocado no acumulador. Essa variável contém a direção do vento. Se for 0, o

vento sopra para a direita; se for 1, para a esquerda. O endereço −5210 contém a posição horizontal da nuvem. Apenas essa posição precisa ser alterada, já que a nuvem se move na mesma linha, de um lado para outro.

O valor de −5207 é testado pela instrução **cp 0**. Se for 0, a instrução **jr z** salta para o rótulo **ct**, o valor em −5210 é incrementado através de HL e a nuvem se move para a direita. Se for 1, o salto não ocorre, o valor em −5210 é decrementado através do par HL e a nuvem se move para a esquerda.

FORMAÇÃO DA NUVEM

Utilizamos sprites para imprimir a nuvem na tela do MSX. No final deste artigo, você terá explicações mais detalhadas sobre o emprego desse recurso gráfico em código de máquina.

Para que esta parte do programa funcione, os padrões das figuras devem estar na memória. Além disso, é preciso que a rotina −12121 — que transfere esses padrões da RAM para a tabela de padrões de sprites na VRAM — tenha sido executada. Se você estiver acompanhando a seqüência do jogo e executar a rotina principal antes da que apresentamos neste artigo, as condições acima serão cumpridas.

A rotina −11168 é utilizada duas vezes para colocar na tela os dois sprites que compõem a nuvem. Assim, os parâmetros necessários precisam ser carregados nos registros adequados.

O par HL deve conter o endereço inicial do sprite na Tabela de Atributos de Sprites (TAS). O endereço inicial dessa tabela está armazenado nas posições 62413 e 62414. Como a nuvem é a primeira figura do jogo que utiliza o sprite, este será o endereço do primeiro sprite da nuvem. O registro B, que deve conter a coordenada Y do sprite na tela, é ajustado com o valor 6. O registro C conterá a coordenada X. Para isso, o conteúdo de −5210 é colocado em C pelo acumulador. O registro D contém o código do padrão de 32 bytes que forma o sprite — no nosso caso, a primeira parte da nuvem tem o código 20. Finalmente, o registro E deve conter a cor do sprite — usamos aqui o preto, cujo código é 1.

Para o sprite que completa nuvem há algumas alterações. Como se trata do segundo sprite do jogo, soma-se 4 ao endereço inicial da TAS no par HL. A coordenada Y em B permanece 6. A coordenada X em C é o resultado da soma do conteúdo anterior desse registro com 16. A operação é necessária porque imprimimos o segundo sprite que compõe a nuvem ao lado direito do primeiro, que tem dezesseis bits de comprimento. O código do segundo sprite da nuvem é 24, valor colocado em D. Sua cor é a mesma — ou seja, o registro E continua contendo o valor 1.

SOPRANDO O VENTO

É importante checar se a nuvem chegou a um dos cantos do vídeo. Caso isso ocorra, estaremos decrementando o valor 0 ou incrementando o valor 255 na coordenada horizontal, o que resultará em erro nessa coordenada.

Verificamos primeiro se o conteúdo de −5210 é 2 ou 230. No primeiro caso, alteramos a direção do vento em −5207 para 0, ou seja, para a direita; no segundo, alteramos para 1, ou seja, para a esquerda. Se o endereço −5210 não contiver nenhum desses dois valores, a direção não é alterada e o processador retorna.

UTILIZAÇÃO DE SPRITES

O sprite é um bloco de 16 X 16 pontos ou de 8 X 8 pontos, dependendo do tamanho selecionado. No primeiro caso, seus padrões ocupam 32 bytes, o que nos permite definir apenas 64 sprites diferentes. No segundo caso, os padrões ocupam oito bytes, como um caractere gráfico comum, e podemos definir até 256 sprites diferentes. Esses padrões devem ser colocados na tabela de padrões de sprites, cujo endereço inicial está armazenado nas posições 62415 e 62416 da RAM.

Os sprites podem ser quatro vezes maiores do que os blocos gráficos comuns, mas não é só isso o que os distingue. Normalmente, para movimentar um caractere gráfico na tela, temos que apagá-lo de sua posição anterior e imprimi-lo na nova posição. Um sprite pode ser movimentado facilmente pela tela: alterando-se suas coordenadas X e Y de impressão, ele será automaticamente apagado da posição que ocupava e impresso na nova posição. Tudo isso é feito pelo processador de vídeo.

Esse recurso gráfico tem, entretanto, suas limitações. Os sprites são definidos numa tabela de 256 bytes, a já mencionada Tabela de Atributos de Sprites — TAS. Os atributos de cada sprite ocupam quatro bytes; portanto, não podemos colocar ao mesmo tempo na tela mais do que 32 sprites diferentes, independentemente do tamanho que tenha sido selecionado.

O primeiro byte guarda a coordenada Y; o segundo, a coordenada X; o terceiro, o número do sprite (é possível escolher entre 64 e 256 padrões diferentes, dependendo do tamanho escolhido); o quarto e último, a cor. Como o sprite tem apenas cor de frente, os bits apagados são transparentes na tela, o que nos permite criar interessantes efeitos visuais.

Uma dúvida pode surgir: o que acontece se as coordenadas de dois ou mais sprites coincidirem? Os sprites são hierarquizados segundo a ordem em que foram definidos na TAS — ou seja, o sprite que ocupa os quatro primeiros bytes na TAS tem precedência sobre o que ocupa os quatro bytes seguintes e assim por diante. Em resumo, o sprite que tem precedência aparecerá na frente dos demais. A justaposição de figuras assim obtida muitas vezes é aproveitada na composição de efeitos visuais. Mas não se esqueça de que só podemos ter até quatro sprites numa mesma linha. Se tentarmos colocar um quinto sprite, o último na ordem hierárquica desaparecerá.

A rotina a seguir coloca na TAS os atributos de um sprite, ou seja, coloca um sprite na tela.

```
10    org -11168
20    ld a,b
30    push de
40    push bc
50    push hl
60    call 77
70    pop hl
80    pop bc
90    inc hl
100    ld a,c
110    push hl
120    call 77
130    pop hl
140    pop de
150    inc hl
160    ld a,d
170    push de
180    push hl
190    call 77
200    pop hl
210    pop de
220    inc hl
230    ld a,e
240    call 77
250    ret
260    end
```

Essa rotina utiliza os parâmetros fornecidos pelo par HL e pelos registros B, C, D e E para colocar na TAS os atributos de um sprite.

O par HL deve conter o endereço inicial do sprite na TAS; o registro B, a coordenada Y; o registro C, a coordenada X; o registro D, o código do sprite e o registro E, a cor do sprite.

CLASSIFICAÇÃO DOS MICROCOMPUTADORES POR LINHA

LINHA	FABRICANTE	MODELO
Apple II +	Appletronica	Thor 2010
Apple II +	CCE	MC-4000 Exato
Apple II +	CPA	Absolutus
Apple II +	CPA	Polaris
Apple II +	Digitus	DGT-AP
Apple II +	Dismac	D-8100
Apple II +	ENIAC	ENIAC II
Apple II +	Franklin	Franklin
Apple II +	Houston	Houston AP
Apple II +	Magnex	DM II
Apple II +	Maxitronica	MX-2001
Apple II +	Maxitronica	MX-48
Apple II +	Maxitronica	MX-64
Apple II +	Maxitronica	Maxitronic I
Apple II +	Microcraft	Craf II Plus
Apple II +	Milmar	Apple II Plus
Apple II +	Milmar	Apple Master
Apple II +	Milmar	Apple Senior
Apple II +	Omega	MC-400
Apple II +	Polymax	Maxxi
Apple II +	Polymax	Poly Plus
Apple II +	Spectrum	Microengenho I
Apple II +	Spectrum	Spectrum ed
Apple II +	Suporte	Venus II
Apple II +	Sycomig	SIC I
Apple II +	Unitron	AP II
Apple II +	Victor do Brasil	Elppa II Plus
Apple II +	Victor do Brasil	Elppa Jr.
Apple IIe	Microcraft	Craft IIe
Apple IIe	Microdigital	TK-3000 IIe
Apple IIe	Spectrum	Microengenho II
MSX	Gradiente	Expert GPC-1
MSX	Sharp	Hotbit HB-8000
Sinclair Spectrum	Microdigital	TK-90X
Sinclair Spectrum	Timex	Timex 2000
Sinclair ZX-81	Apply	Apply 300
Sinclair ZX-81	Engebras	AS-1000
Sinclair ZX-81	Filcres	NEZ-8000
Sinclair ZX-81	Microdigital	TK-82C
Sinclair ZX-81	Microdigital	TK-83
Sinclair ZX-81	Microdigital	TK-85
Sinclair ZX-81	Prologica	CP-200
Sinclair ZX-81	Ritas	Ringo R-470
Sinclair ZX-81	Timex	Timex 1000
Sinclair ZX-81	Timex	Timex 1500
TRS-80 Mod. I	Dismac	D-8000
TRS-80 Mod. I	Dismac	D-8001/2
TRS-80 Mod. I	LNW	LNW-80
TRS-80 Mod. I	Video Genie	Video Genie I
TRS-80 Mod.III	Digitus	DGT-100
TRS-80 Mod.III	Digitus	DGT-1000
TRS-80 Mod.III	Kemitron	Naja 800
TRS-80 Mod.III	Prologica	CP-300
TRS-80 Mod.III	Prologica	CP-500
TRS-80 Mod.III	Sysdata	Sysdata III
TRS-80 Mod.III	Sysdata	Sysdata Jr.
TRS-80 Mod.IV	Multix	MX-Compacto
TRS-80 Mod.IV	Sysdata	Sysdata IV
TRS-Color	Codimex	CS-6508
TRS-Color	Dynacom	MX-1600
TRS-Color	LZ	Color 64
TRS-Color	Microdigital	TKS-800
TRS-Color	Prologica	CP-400

CLASSIFICAÇÃO DOS MICROCOMPUTADORES POR NOME

FABRICANTE	MODELO	PAÍS	LINHA
Appletronica	Thor 2010	Brasil	Apple II +
Apply	Apply 300	Brasil	Sinclair ZX-81
CCE	MC-4000 Exato	Brasil	Apple II +
CPA	Absolutus	Brasil	Apple II +
CPA	Polaris	Brasil	Apple II +
Codimex	CS-6508	Brasil	TRS-Color
Digitus	DGT-100	Brasil	TRS-80 Mod.III
Digitus	DGT-1000	Brasil	TRS-80 Mod.III
Digitus	DGT-AP	Brasil	Apple II +
Dismac	D-8000	Brasil	TRS-80 Mod. I
Dismac	D-8001/2	Brasil	TRS-80 Mod. I
Dismac	D-8100	Brasil	Apple II +
Dynacom	MX-1600	Brasil	TRS-Color
ENIAC	ENIAC II	Brasil	Apple II +
Engebras	AS-1000	Brasil	Sinclair ZX-81
Filcres	NEZ-8000	Brasil	Sinclair ZX-81
Franklin	Franklin	USA	Apple II +
Gradiente	Expert GPC1	Brasil	MSX
Houston	Houston AP	Brasil	Apple II +
Kemitron	Naja 800	Brasil	TRS-80 Mod.III
LNW	LNW-80	USA	TRS-80 Mod. I
LZ	Color 64	Brasil	TRS-Color
Magnex	DM II	Brasil	Apple II +
Maxitronica	MX-2001	Brasil	Apple II +
Maxitronica	MX-48	Brasil	Apple II +
Maxitronica	MX-64	Brasil	Apple II +
Maxitronica	Maxitronic I	Brasil	Apple II +
Microcraft	Craft II Plus	Brasil	Apple II +
Microcraft	Caft IIe	Brasil	Apple IIe
Microdigital	TK-3000 IIe	Brasil	Apple IIe
Microdigital	TK-82C	Brasil	Sinclair ZX-81
Microdigital	TK-83	Brasil	Sinclair ZX-81
Microdigital	TK-85	Brasil	Sinclair ZX-81
Microdigital	TK-90X	Brasil	Sinclair Spectrum
Microdigital	TKS-800	Brasil	TRS-Color
Milmar	Apple II Plus	Brasil	Apple II +
Milmar	Apple Master	Brasil	Apple II +
Milmar	Apple Senior	Brasil	Apple II +
Multix	MX-Compacto	Brasil	TRS-80 Mod.IV
Omega	MC-400	Brasil	Apple II +
Polymax	Maxxi	Brasil	Apple II +
Polymax	Poly Plus	Brasil	Apple II +
Prologica	CP-200	Brasil	Sinclair ZX-81
Prologica	CP-300	Brasil	TRS-80 Mod.III
Prologica	CP-400	Brasil	TRS-Color
Prologica	CP-500	Brasil	TRS-80 Mod.III
Ritas	Ringo R-470	Brasil	Sinclair ZX-81
Sharp	Hotbit HB-8000	Brasil	MSX
Spectrum	Microengenho I	Brasil	Apple II +
Spectrum	Microengenho II	Brasil	Apple IIe
Spectrum	Spectrum ed	Brasil	Apple II +
Suporte	Venus II	Brasil	Apple II +
Sycomig	SIC I	Brasil	Apple II +
Sysdata	Sysdata III	Brasil	TRS-80 Mod.III
Sysdata	Sysdata IV	Brasil	TRS-80 Mod.IV
Sysdata	Sysdata Jr.	Brasil	TRS-80 Mod.III
Timex	Timex 1000	USA	Sinclair ZX-81
Timex	Timex 1500	USA	Sinclair ZX-81
Timex	Timex 2000	USA	Sinclair Spectrum
Unitron	AP II	Brasil	Apple II +
Victor do Brasil	Elppa II Plus	Brasil	Apple II +
Victor do Brasil	Elppa Jr.	Brasil	Apple II +
Video Genie	Video Genie I	USA	TRS-80 Mod. I

UM LOGOTIPO PARA CADA MODELO DE COMPUTADOR

INPUT foi especialmente projetado para microcomputadores compatíveis com as sete principais linhas existentes no mercado.
Os blocos de textos e listagens de programas aplicados apenas a determinadas linhas de micros podem ser identificados por meio dos seguintes símbolos:


Sinclair ZX-81


TRS-80


TK-2000


MSX


Spectrum


TRS-Color


Apple II

Quando o emblema for seguido de uma faixa, então tanto o texto como os programas que se seguem passam a ser específicos para a linha indicada.

NOVOS LANÇAMENTOS, NOVOS SUCESSOS.

JÁ NAS LIVRARIAS

A FRATERNIDADE DA PEDRA
David Morrell

Um grupo secreto, sob direção de um padre armado, passa a agir contra o terrorismo. Mas, será que violência se combate com mais violência? Eis o dilema de Drew, agente da Lei envolvido com fatos e figuras do mundo real, num livro surpreendente do criador de Rambo.

AVENTUREIROS E MILIONÁRIOS
Clark Howard

No romance do Texas, a saga da descoberta de um poço de petróleo e o drama de um casal que herda um lote de terra aparentemente sem valor e enfrenta com coragem os poderosos do lugar, até vencer. Uma vitória antes de tudo moral, numa história forte e envolvente, com realistas cenas de amor.

CAÇADA SEM FIM
Bryan Forbes

Uma brilhante história de espionagem envolvendo a KGB. Por que matar uma ex-espiã que já tinha sido desmascarada e torturada tempos atrás? Um agente inglês, seu antigo amante, enfrenta um desafio: descobrir por que ela foi morta... e por que *agora!*

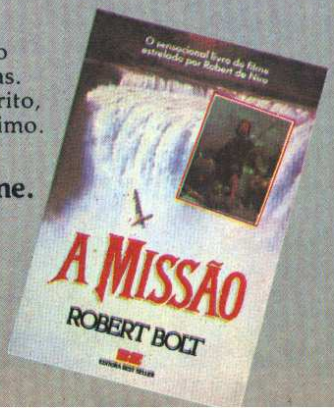
PODER
Howard Fast

Um líder sindical com a volúpia do poder, a luta pelos direitos dos trabalhadores, nos Estados Unidos, e sua manipulação por corruptos e oportunistas; o jogo das ambições políticas. Admiravelmente escrito, um romance atualíssimo.

SUPERSEXO
Alexandra Penney

Se não for o primeiro, este vai ser o último e definitivo guia para o prazer que o leitor poderá seguir: um livro que derruba mitos, faz sugestões provocantes e propõe técnicas ousadas para se chegar ao supersexo, uma relação intensa e especial entre os casais, que não exclui o romantismo.

Não perca também: A MISSÃO, de Robert Bolt, o livro do filme.



EDITORA BEST SELLER