CURSO PRÁTICO 57 DE PROGRAMAÇÃO DE COMPUTADORES

INPUT

PROGRAMAÇÃO BASIC - PROGRAMAÇÃO DE JOGOS - CÓDIGO DE MÁQUINA

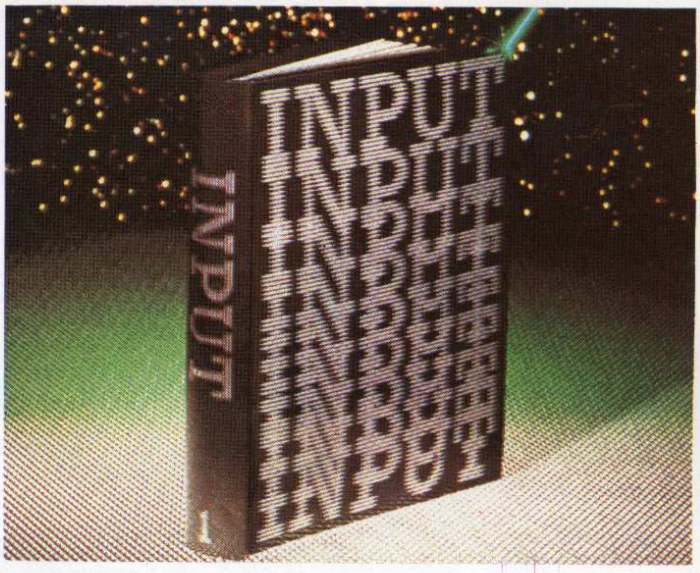
NOVA CULTURAL

Cz$ 50,00

INPUT

Vol. 4 N.º 57

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 nº (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.

Editor
VICTOR CIVITA

REDAÇÃO
Diretor Editorial: Carmo Chagas

Editores Executivos: Antonio José Filho, Berta Sztark Amar

Editor Chefe: Paulo de Almeida
Editora de Texto: Ana Lúcia B. de Lucena
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

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

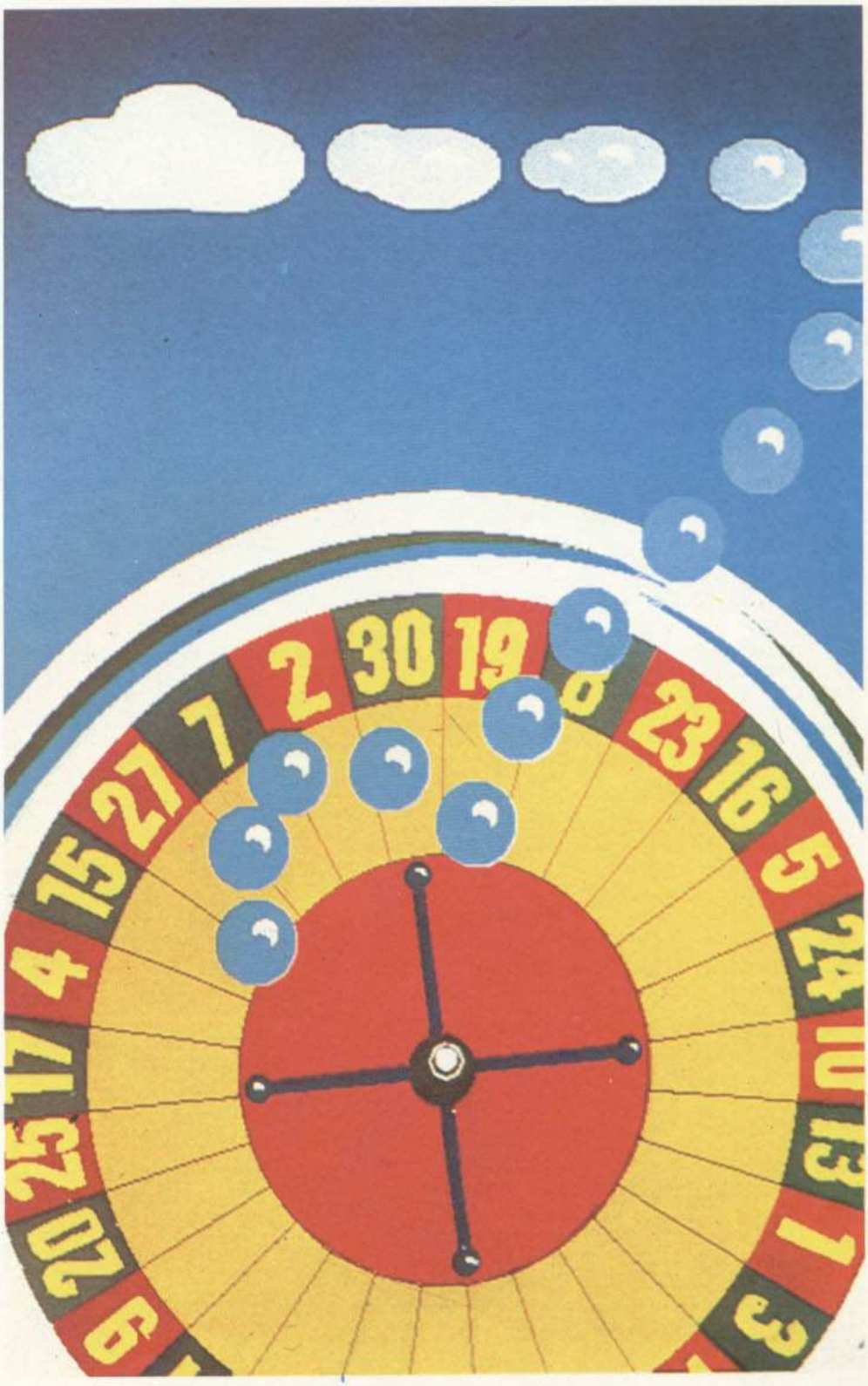
Preparador de Texto/Coordenador: Eliel Silveira Cunha
Preparadores de Texto: Alzira Moreira Braz, Ana Maria Dilguerian, Levon Yacubian, Luciano Tasca, Maria Teresa Galluzzi, Maria Teresa Martins Lopes, Paulo Felipe Mendrone
Revisor/Coordenador: José Maria de Assis
Revisoras: Conceição Aparecida Gabriel, 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, nº 2000 - 3º 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.

SIMULAÇÃO E PREVISÃO

- FUNDAMENTOS E APLICAÇÕES
- SIMULAÇÃO DE UMA LOTERIA
- GERAÇÃO DE NÚMEROS RANDÔMICOS
- AMOSTRAGEM E PESQUISA

Quais são suas chances de fazer os treze pontos na loteria esportiva desta semana? Nosso programa não vai melhorar sua sorte, mas poderá ajudá-lo a entender porque perdeu.



Instruir um novo piloto a bordo de um avião a jato ou de um bombardeiro pode custar muito caro, se levarmos em conta os custos com combustível, aterrissagens e pessoal de apoio. Por isso, as autoridades militares preferem investir grandes somas em aviões de treinamento e simuladores — equipamentos controlados por computador que nunca deixam o solo, mas dão ao iniciante a sensação de um vôo real.

O mesmo tipo de argumento é válido em várias outras situações — incluindo o planejamento de uma indústria, marketing, pesquisas científicas e planos governamentais. Programar um computador para simular os efeitos de uma provável falência ou de uma determinada política econômica é, sem dúvida, mais razoável do que testá-las na prática, o que tomaria cinco anos ou mais, dependendo das circunstâncias. Não surpreendem, portanto, os grandes investimentos que têm sido feitos na área.

Antes do advento dos computadores, era praticamente impossível utilizar esse recurso, devido à complexidade e extensão dos cálculos envolvidos. Hoje em dia, até os microcomputadores pessoais podem fazer simulações não muito complicadas — como as que você já deve ter visto em jogos comercializados pelos fabricantes.

Por trás de todo trabalho de simulação e previsão, estão as leis da probabilidade. Se você não está familiarizado com elas, consulte o artigo *Acaso e Probabilidade* na página 776. Complementando essas leis, usam-se os estudos estatísticos, que analisam os fatos reais, quantificando a ocorrência de certas situações e o surgimento de variações em suas características.

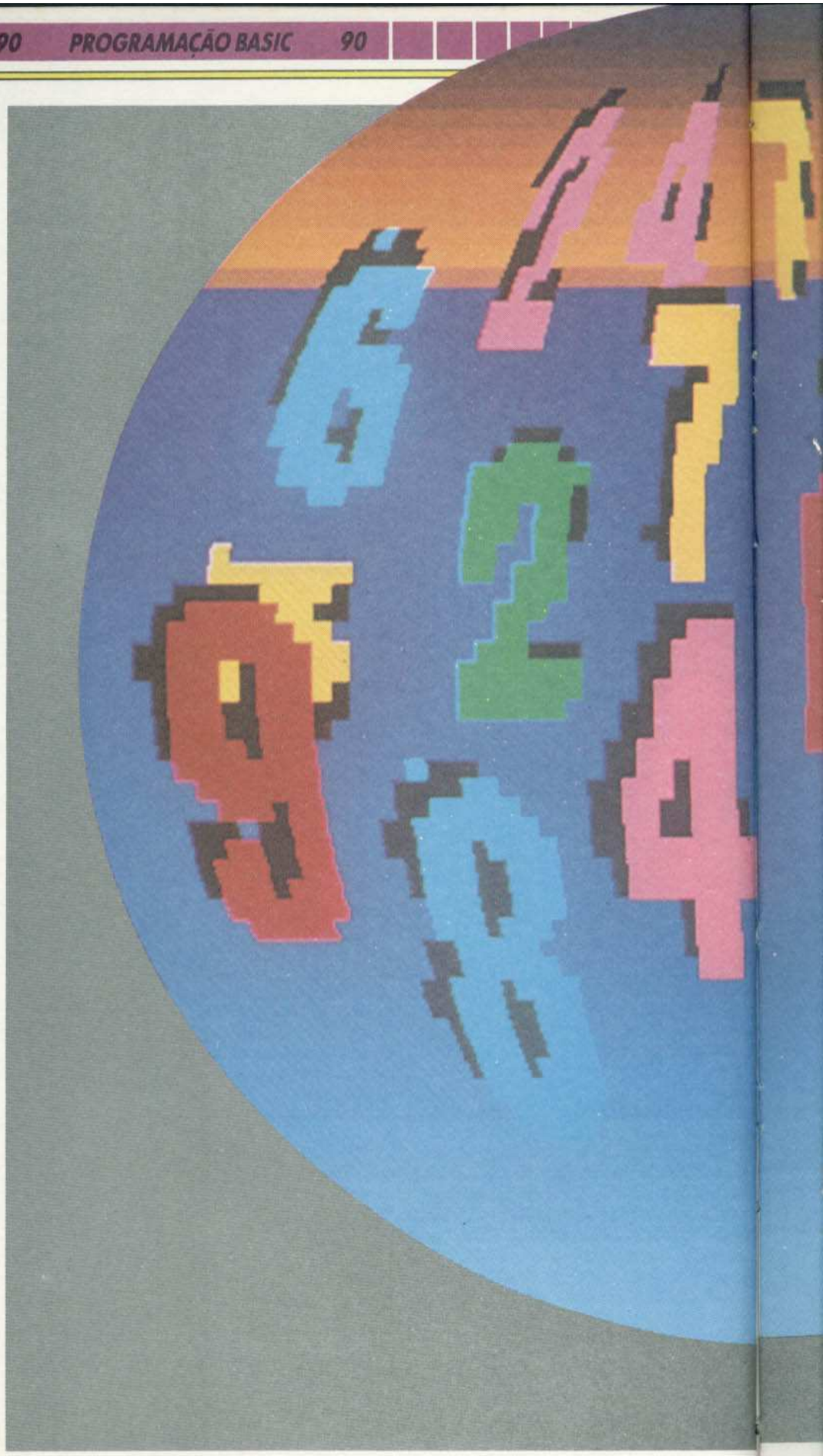
A partir de algumas informações básicas, o computador é capaz de prever situações futuras para uma série de eventos. Mas a confiabilidade dos resul-

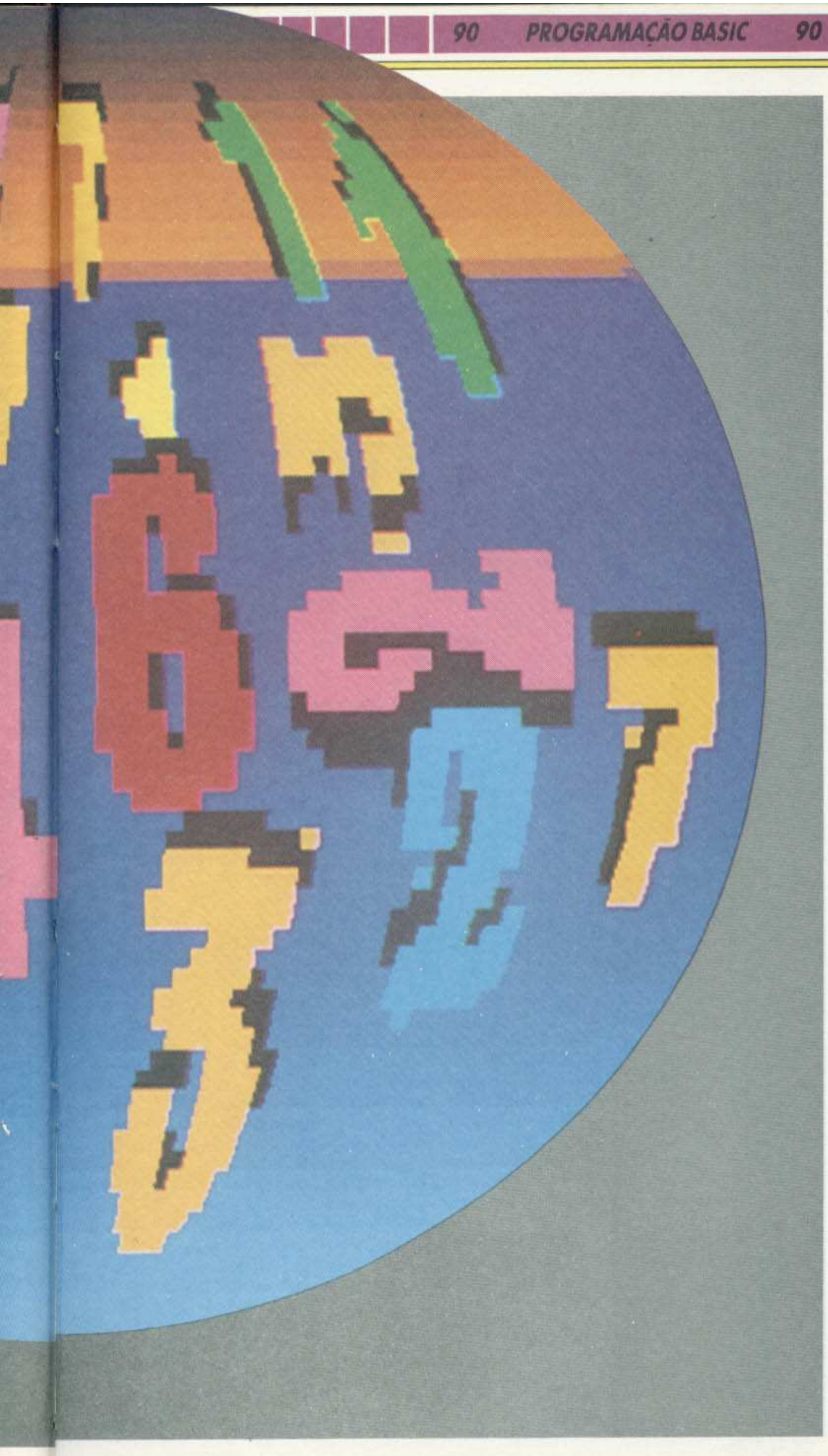
tados dependerá sempre do cuidado com que os dados foram introduzidos e da precisão das regras que orientam sua manipulação. Tomemos um exemplo bem simples: a simulação dos resultados de uma loteria esportiva.

A cada fim de semana, milhares de pessoas conferem, pela televisão ou pelo rádio, os resultados da loteria esportiva, aguardando ansiosamente sua vez de fazer os treze pontos. Infelizmente, todos, exceto os raros premiados, desligam a televisão desapontados e vão dormir, esperando pelos jogos da próxima semana. Se voçê quiser antecipar a emoção de checar sua sorte, digite estas linhas:

```
 10 DIM J(13): BORDER 0: PAPER
  0: INK 7: CLS
 20 PRINT  INVERSE 1;" AS PART
 IDAS ESTAO EM ANDAMENTO"
 30 FOR I=1 TO 13
 40 LET J(I)=RND*1
 50 NEXT I
 60 FOR I=1 TO 2000: NEXT I
 70 PRINT : PRINT  FLASH 1;
 PAPER 2;" E ATENCAO PARA OS R
 ESULTADOS:": PRINT
 80 FOR I=1 TO 13
 90 IF J(I)<=.5 THEN  LET Z$="
 UM"
100 IF J(I)>.5 AND J(I)<=.75
THEN  LET Z$="DOIS"
110 IF J(I)>.75 THEN  LET Z$="
DO MEIO"
120 PRINT TAB 3;"JOGO ";I;:
PRINT TAB 11;": COLUNA ";Z$
130 FOR T=1 TO 500: NEXT T
140 NEXT I: PRINT : PRINT
150 PRINT  FLASH 0; PAPER 1;"V
OCE QUER OUTRA LOTERIA (S/N)?"
160 LET A$=INKEY$: IF A$=""
THEN  GOTO 160
170 IF A$="S" THEN  GOTO 10
180 STOP
```

```
10 DIM J(13)
20 CLS:PRINT " AS PARTIDAS ESTA
O EM ANDAMENTO"
30 FOR I=1 TO 13
40 LET J(I)=RND(10)/10
50 NEXT I
60 FOR I=1 TO 2000:NEXT
70 PRINT @34,"E ATENCAO PARA OS
 RESULTADOS":PRINT
80 FOR I=1 TO 13
90 IF J(I)<=.5 THEN Z$="UM"
100 IF J(I)>.5 AND J(I)<=.75 TH
EN Z$="DOIS"
110 IF J(I)>.75 THEN Z$="DO MEI
O"
120 PRINT TAB(4) "JOGO";I;": CO
LUNA ";Z$
130 FOR T=1 TO 500:NEXT T
140 NEXT I:PRINT
150 PRINT "VOCE QUER OUTRA LOTE
```





```
RIA ? (S/N)"
160 LET A$=INKEY$:IF A$="" THEN
 160
170 IF A$="S" THEN GOTO 20
180 END
```

```
10  DIM J(13)
20  HOME : PRINT  TAB( 5)"AS PA
RTIDAS ESTAO EM ANDAMENTO"
30  FOR I = 1 TO 13
40  LET J(I) =  RND (1)
50  NEXT I
60  FOR I = 1 TO 2000: NEXT
70  PRINT : PRINT  TAB( 6)"E AT
ENCAO PARA OS RESULTADOS": PRIN
T
80  FOR I = 1 TO 13
90  IF J(I) <  = .5 THEN Z$ = "
UM"
100  IF J(I) > .5 AND J(I) <  =
 .75 THEN Z$ = "DOIS"
110  IF J(I) > .75 THEN Z$ = "D
O MEIO"
120  PRINT  TAB( 7)"JOGO ";I;:
PRINT  TAB( 15)" :   COLUNA ";Z
$
130  FOR T = 1 TO 500: NEXT T
140  NEXT I: PRINT : PRINT
150  PRINT  TAB( 4)"VOCE QUER O
UTRA LOTERIA ?(S/N)"
160  GET A$
170  IF A$ = "S" THEN  GOTO 20
180  END
```

```
10 DIMJ(13)
20 CLS:PRINT TAB(5)"AS PARTIDAS
 ESTAO EM ANDAMENTO"
30 FOR I=1 TO 13
40 LET J(I)=RND(1)
50 NEXT I
60 FOR I=1 TO 2000:NEXT
70 PRINT:PRINT TAB(6) "E ATENÇA
O PARA OS RESULTADOS":PRINT
80 FOR I=1 TO 13
90 IF J(I)<=.5 THEN Z$="UM"
100 IF J(I)>.5 AND J(I)<=.75 TH
EN Z$="DOIS"
110 IF J(I)>.75 THEN Z$="DO MEI
O"
120 PRINT TAB(7) "JOGO";I;:PRIN
T TAB(15)" :   COLUNA ";Z$
130 FOR T=1 TO 500:NEXT T
140 NEXT I:PRINT:PRINT
150 PRINT TAB(4) "VOCÊ QUER OUT
RA LOTERIA? (S/N)"
160 A$=INKEY$:IF A$="" THEN GOT
O 160
170 IF A$="S" THEN GOTO 20
180 END
```

Rode o programa e os resultados serão prontamente impressos na tela.

Não é incomum o cancelamento de algum jogo, sobretudo por causa de chuva. Quando isso ocorre, a Caixa Econômica Federal, a promotora da loteria, faz um sorteio para definir um resultado para o jogo. Na verdade, esse processo não passa de uma simulação,

ou seja, de uma representação simbólica de uma situação real.

A estrutura do programa é muito simples. O laço entre as linhas 30 e 50 sorteia treze números correspondentes aos treze jogos, e o laço entre as linhas 80 e 140 relaciona esses números aos possíveis resultados.

DECIDINDO OS RESULTADOS

O parâmetro que o computador utiliza para imprimir os dados do placar provém de uma análise sobre a freqüência de cada resultado em jogos passados. Sabe-se que, em 50% dos jogos, o time da casa vence, pois conta, entre outras vantagens, com o apoio da torcida. As ocorrências restantes dividem-se igualmente entre empate — 25% — e vitória do time visitante — 25%.

Em cada jogo há três resultados possíveis (coluna um, coluna do meio e coluna dois) e a loteria inclui treze jogos. Assim, a probabilidade de ocorrer determinado resultado é de 1 em 1.594.323 (um em três elevado a treze). Como você vê, as chances de acerto seriam muito pequenas, caso não se permitissem apostas duplas ou triplas. Na Inglaterra, por exemplo, as apostas também incluem empate com gols ou sem gols, o que torna a probabilidade de ganhar ainda menor (1 em 4↑13).

Decidir o resultado de um jogo é, portanto, como fazer um sorteio: os números são colocados em uma caixa e alguém, sem olhar, pega um deles.

Suponhamos que se divida um papel em quatro partes: na primeira, escrevemos "coluna dois"; na segunda, "coluna do meio" e, nas outras duas, "coluna um". Se fizermos um sorteio, estaremos representando a decisão de uma partida de futebol. Também poderíamos escrever um número em cada pedaço de papel — 1, 2, 3 e 4 — e relacioná-los a um resultado desta forma:

NÚMERO	RESULTADO
1, 2	coluna um
3	coluna dois
4	coluna do meio

A função RND do BASIC, que gera números randômicos, faz com que o computador sorteie um número do "chapéu". Considerando que **RND** gera um número entre 0 e 1, podemos reescrever nossa tabela assim:

VALOR DO RND	RESULTADO
de 0 a 0.5	coluna um
maior que 0.5 até 0.75	coluna dois
maior que 0.75 até 1.00	coluna do meio

Voltando ao programa anterior, você pode observar que este é o processo utilizado pelo computador para decidir o resultado de cada jogo (linha 80 até linha 140).

Em nosso programa, as proporções entre os resultados dos jogos foram estimadas um pouco grosseiramente. Para aperfeiçoá-las, você poderá fazer uma pequena pesquisa em arquivos de jornais e redefinir a proporção a ser usada. Experimente fazer algumas previsões e confira com os resultados da próxima semana. Boa sorte!

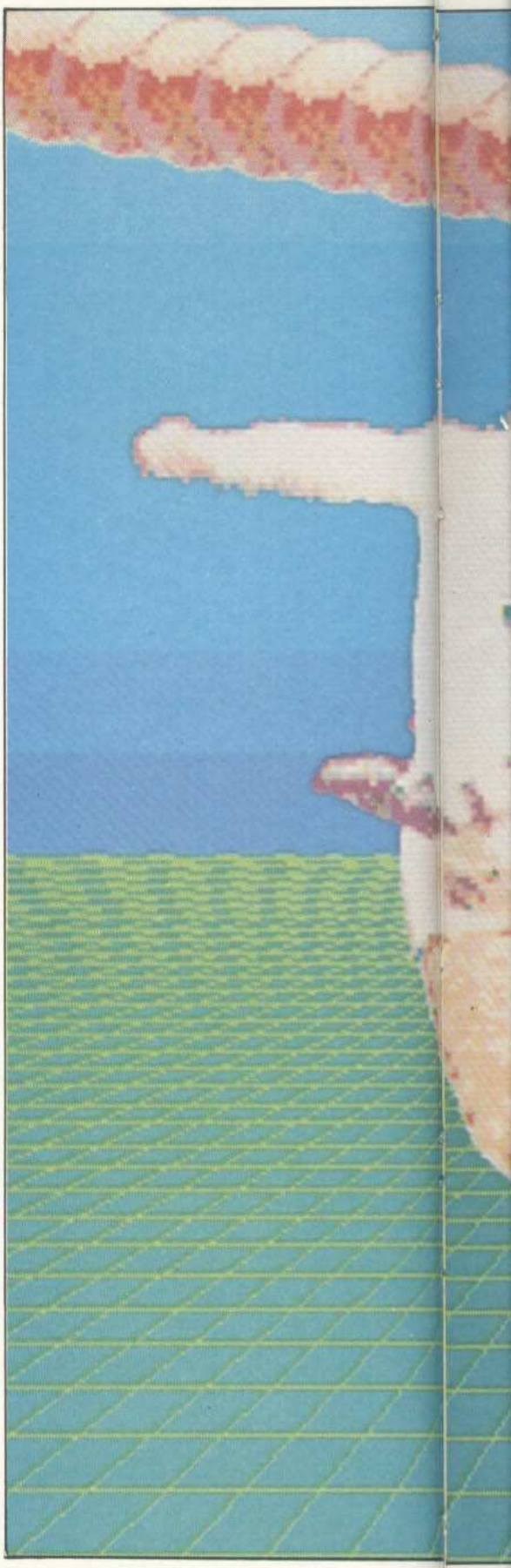
GERAÇÃO DE NÚMEROS RANDÔMICOS

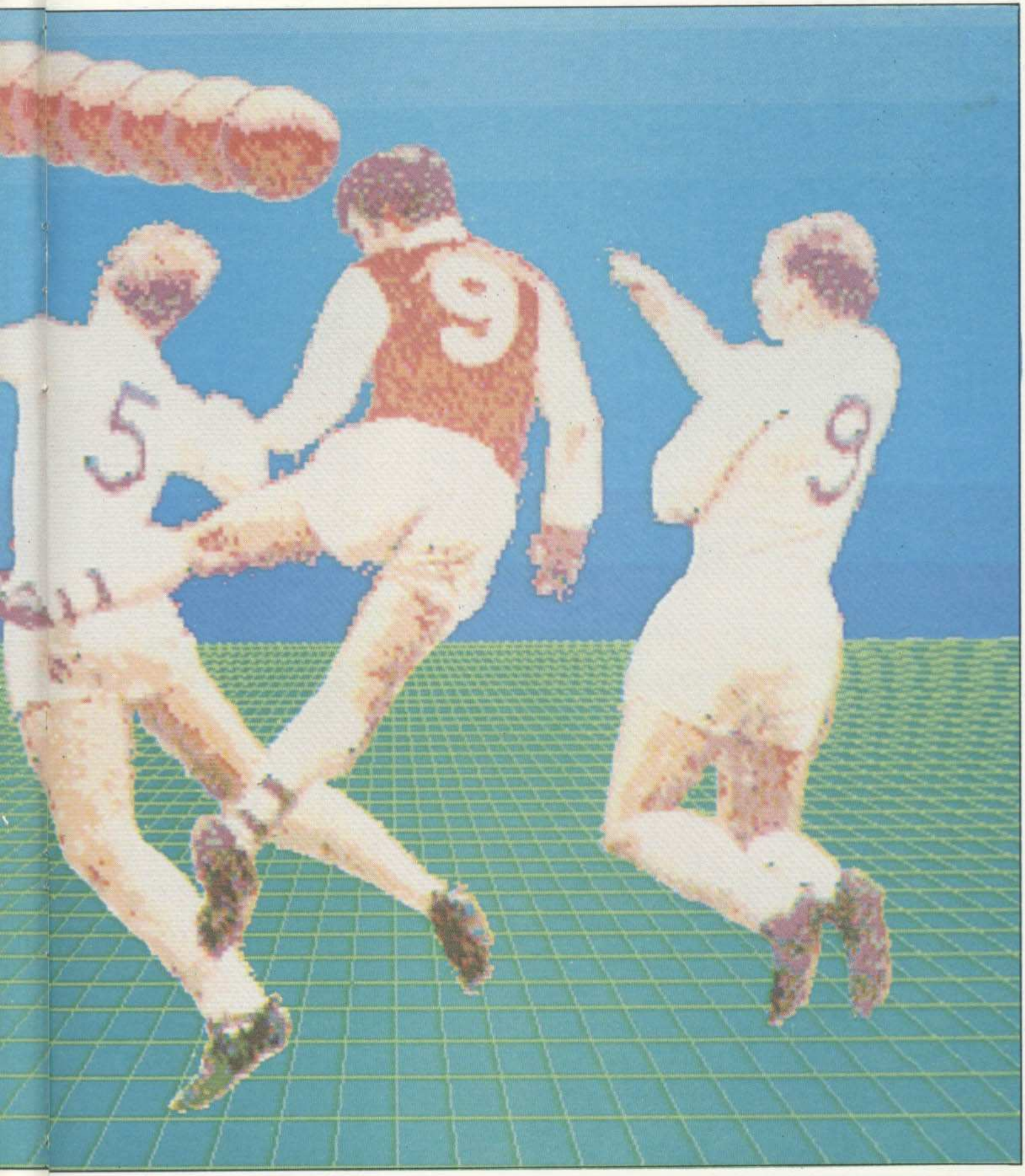
Dados, cartas e roleta já foram muito utilizados para a obtenção de números randômicos. Procedimentos como esses, lentos e tediosos, puderam ser abandonados depois que um famoso matemático americano, John von Newman, propôs o método da potência quadrada. Começando com um número de quatro dígitos (a "semente"), o próximo número randômico seria obtido pela multiplicação da semente por ela mesma. Do resultado seriam destacados os quatro dígitos do meio. Suponhamos que a semente é o número 5272. O segundo número será gerado pelos quatro dígitos centrais de (5272↑2), ou seja, 27.793.984. A resposta (7939) é praticamente randômica. Um terceiro número seria obtido elevando-se 7939 ao quadrado e assim por diante.

Você pode estar se perguntando se qualquer processo matemático — obrigatoriamente repetitivo — seria capaz de gerar números realmente randômicos. Na verdade, isso é impossível. Porém, o número obtido comporta-se como se fosse randômico, e é geralmente chamado pseudo-randômico.

Infelizmente, a técnica da potência quadrada não é muito útil para gerar números randômicos no computador. Além de ser um processo lento, a seqüência logo se repete, e, assim que surge um zero, ela é interrompida. A maioria dos micros adota um método distinto, recorrendo a uma fonte qualquer de números para gerar os pseudo-randômicos. O próximo programa usa uma fórmula bem simples e a função **INT** para demonstrar como esse método funciona.

```
20 BORDER 0: INK 7: PAPER 0:
CLS
30 PRINT AT 0,2; INVERSE 1;"
NUMEROS PSEUDO-RANDOMICOS "''
40 INPUT " QUANTOS NUMEROS ?
";n: LET s=0
```





Quantas variáveis cabem na memória?

Nos interpretadores Microsoft BASIC dos micros MSX, TRS-80, TRS-Color, Apple e TK-2000, os nomes de variáveis podem ter uma ou duas letras, ou uma letra e um número. Também é possível acrescentar o sufixo de tipo, que pode ser um # (precisão dupla), ! (precisão simples), % (inteiro) e $ (literal). Dadas essas limitações, calcula-se o número de nomes diferentes em 2 000.

Já para os micros Sinclair (ZX-81 e Spectrum), não há limite teórico, pois são aceitos mais de dois caracteres no nome de uma variável.

```
50 LET  x =.677829*PEEK 23673
/50
60 LET x=x*1842.95
70 LET x=x-INT (x)
80 LET p=INT (x*1000): PRINT
"     ";.001*p,
90 LET s=s+1
110 IF s<=n THEN  GOTO 60
120 STOP
```

```
20 CLS
30 PRINT @3,"NUMEROS PSEUDO-RAN
DOMICOS"
40 PRINT:PRINT:INPUT"QUANTOS NU
MEROS ";N:S=0
50 X=.677829*TIMER/50
60 X=X*1842.95
70 X=X-INT(X)
80 P=INT(X*1000):PRINT LEFT$(ST
R$(P/1000)+"        ",8);
90 S=S+1
110 IF S<=N THEN 60
120 END
```

```
20   HOME
30   PRINT  TAB( 7)"NUMEROS PSEU
DO-RANDOMICOS": PRINT
40   INPUT "QUANTOS NUMEROS ? ";
N:S = 0
50 X = .677829 * 318378 / 50
60 X = X * 1842.95
70 X = X -  INT (X)
80 P =  INT (X * 1000): PRINT
LEFT$ ( STR$ (P / 1000) + "
    ",8);
90 S = S + 1
110  IF S < N THEN 60
120  END
```

```
20 CLS
30 PRINT TAB(6) "NUMEROS PSEUDO
-RANDOMICOS":PRINT
40 INPUT"QUANTOS NUMEROS ";N:S=
0
50 X=.677829*TIME/50
60 X=X*1842.95
70 X=X-INT(X)
80 P=INT(X*1000):PRINT LEFT$(ST
R$(P/1000)+"        ",8);
90 S=S+1
110 IF S<N THEN 60
120 END
```

A linha 50 usa a função *time* para produzir uma semente diferente a cada execução do programa. O Apple e o TK-2000 não possuem essa função. Portanto, se você quiser uma outra seqüência, deverá substituir o valor 318378.

O número .677829 é uma constante arbitrária. Depois que o valor é multiplicado por outra constante (linha 60), obtém-se o resíduo (ou parte decimal). Mude os valores das constantes nas linhas 50 e 60 e torne a executar o programa para verificar o tipo de resultado a que chegará.

Para outros fins, é mais fácil utilizar a função **RND** existente em seu computador. Variando o valor de **x** na expressão **RND(x)**, você poderá selecionar uma seqüência que se repita, o que é muito útil para renovar a semente a cada execução do programa.

Lembre-se de que a função **RND** cria uma variável randômica e não uma variável algébrica. **RND(1)** — ou **RND(0)** no TRS-Color — não resultará no mesmo número em duas partes do programa.

AMOSTRAS E PESQUISAS

Empresas que fazem pesquisas de mercado ou de opinião pública usam computadores para gerar uma amostra randômica da população. Quando entrevistam, por exemplo, mil pessoas, é muito importante que elas representem a grande maioria da população. Ou seja, os entrevistados não podem ser escolhidos por morar perto do instituto de pesquisa ou trabalhar em determinada empresa. Ao contrário, devem ter as mais diversas origens e características para que suas opiniões não sejam influenciadas por particularidades comuns a determinados grupos.

A melhor maneira de se definir uma amostra bem representativa é selecioná-la randomicamente, excluindo, assim, qualquer tendência ou preconceito.

Isso continua valendo mesmo que o campo amostral não seja tão grande, e que comporte de fato algumas características comuns — como se supõe haver entre os sócios de um clube ou os leitores de INPUT. Se você quisesse saber, por exemplo, quais os computadores mais usados entre os leitores de INPUT, seria necessário fazer uma amostragem randômica para chegar ao resultado mais próximo possível do real.

O processo utilizado no programa é similar ao de retirar números de um chapéu, com uma pequena diferença: o papel sorteado não voltará ao chapéu, o que equivaleria a entrevistar duas vezes a mesma pessoa. Execute o programa que se segue para ver como a função **RND** pode ser empregada para gerar uma amostragem randômica.

```
 10 DIM b$(10,16)
 20 DIM a$(10,16)
 30 BORDER 0: PAPER 0: INK 7:
 CLS
 50 PRINT AT 0,5; INVERSE 1;"A
 MOSTRAGEM RANDOMICA "
 90 PAUSE 100: CLS
100 FOR i=1 TO 10: READ a$(i):
 NEXT i
110 INPUT " TAMANHO DA AMOSTRA
 ? ";n
120 FOR v=1 TO 10: LET b$(v)=a
$(v): NEXT v
130 FOR j=1 TO n
140 LET r=1+INT (RND*10)
150 IF b$(r)="
" THEN  GOTO 140
160 PRINT b$(r)
170 LET b$(r)=""
180 NEXT j
190 INPUT " OUTRA AMOSTRA (s/n
) ? ";q$
200 IF q$="s" THEN  CLS : GOTO
 110
210 STOP
220 DATA "BONN","COPENHAGUE","
LONDRES"
230 DATA "MADRI","MOSCOU","NOV
A IORQUE"
240 DATA "PARIS","ROMA","ESTOC
OLMO","VIENA"
```

```
40 CLS
50 PRINT @8,"AMOSTRA RANDOMICA"
:PRINT:PRINT
100 RESTORE:FOR I=1 TO 10:READ
A$(I):NEXT I
110 INPUT"TAMANHO DA AMOSTRA ";
N:PRINT
115 IF N>10 THEN 40
120 FOR V=1 TO 10:B$(V)=A$(V):N
EXT V
130 FOR J=1 TO N
140 R=1+INT(RND(0)*10)
150 IF B$(R)="" THEN 140
160 PRINT B$(R)
170 B$(R)=""
180 NEXT J
190 PRINT:INPUT" OUTRA AMOSTRA
(S/N) ";G$:PRINT:PRINT
200 IF G$="S" THEN 110
210 END
220 DATA BONN,COPENHAGUE,LONDRE
```

```
S
230 DATA MADRI,MOSCOU,NOVA IORQ
UE
240 DATA PARIS,ROMA,ESTOCOLMO,V
IENA
```

```
40  HOME
50  PRINT  TAB( 10)"AMOSTRAGEM
RANDOMICA": PRINT
100  RESTORE : FOR I = 1 TO 10:
 READ A$(I): NEXT I
110  INPUT "TAMANHO DO CAMPO AM
OSTRAL :";N: PRINT
115  IF N > 10 THEN 40
120  FOR V = 1 TO 10:B$(V) = A$
(V): NEXT V
130  FOR J = 1 TO N
140 R = 1 +  INT ( RND (1) * 10
)
150  IF B$(R) = "" THEN 140
160  PRINT B$(R)
170 B$(R) = ""
180  NEXT J
190  PRINT : INPUT "OUTRA AMOST
RAGEM ?(S/N) ";G$: PRINT : PRIN
T
200  IF G$ = "S" THEN 40
210  END
220  DATA  BONN,COPENHAGEN,LOND
RES
230  DATA  MADRI,MOSCOU,NOVA YO
RK
240  DATA   PARIS,ROMA,ESTOCOLM
O,VIENA
```

```
20 CLS
30 PRINT TAB(6) "NUMEROS PSEUDO
-RANDOMICOS":PRINT
40 CLS
50 PRINT TAB(8) "AMOSTRAGEM RAN
DOMICA":PRINT:PRINT
100 RESTORE:FOR I=1 TO 10:READ
A$(I):NEXT I
110 INPUT "TAMANHO DA AMOSTRA "
;N:PRINT
120 FOR V=1 TO 10:B$(V)=A$(V):N
EXT V
130 FOR J=1 TO N
140 R=1 +INT(RND(1)*10)
150 IF B$(R)="" THEN 140
160 PRINT B$(R)
170 B$(R)=""
180 NEXT J
190 PRINT:INPUT"OUTRA AMOSTRA ?
 (S/N)";G$:PRINT:PRINT
200 IF G$="S" THEN 110
210 END
220 DATA BONN,COPENHAGEN,LONDRE
S
230 DATA MADRI,MOSCOU,NOVA YORK
240 DATA PARIS,ROMA,ESTOCOLMO,V
IENA
```

Depois que as informações que seguem o comando **DATA** foram lidas (linha 100), um número inteiro randômico **R**, entre 1 e 10, é gerado (linha 140). Seu computador imprimirá, então, o **R**-ésimo item (linha 160) da lista.

Assim que um item é selecionado, deve-se removê-lo do banco de informações, para não ser escolhido duas vezes. A linha 170 se encarrega disso.

Geralmente, o banco de informações a ser pesquisado é bem mais extenso do que o apresentado aqui. Para obter amostragens de um grupo maior, nosso programa seria lento e ineficiente. Se um político quisesse, por exemplo, sortear duzentos nomes de um eleitorado de 60.000 elementos, o programa teria que percorrer a lista duzentas vezes. Para evitar essa longa espera, o mais conveniente seria recorrer ao método de busca individual.

O MÉTODO DE BUSCA INDIVIDUAL

Com esse método, o banco de dados é percorrido uma só vez, de cima para baixo. Decide-se, a cada nome considerado, qual deve ser incluído na amostragem. Para obter uma amostragem dessa maneira, faça as seguintes modificações no programa anterior:

```
120 LET a=n: LET c=10
130 FOR j=1 TO 10
140 IF a=0 THEN  GOTO 190
150 IF RND*1<=a/c THEN  PRINT
a$(j): GOTO 170
160 LET c=c-1: GOTO 180
170 LET a=a-1: LET c=c-1
```

```
50 PRINT @3,"AMOSTRA RANDOMICA
SIMPLES ":PRINT:PRINT
120 A=N:C=10
130 FOR J=1 TO 10
140 IF A=0 THEN 190
150 IF RND(0)<=A/C THEN PRINT A
$(J):GOTO 170
160 C=C-1:GOTO 180
170 A=A-1:C=C-1
```

```
50  PRINT "AMOSTRAGEM RANDOMICA
 DE BUSCA INDIVIDUAL": PRINT
120 A = N:C = 10
130  FOR J = 1 TO 10
140  IF A = 0 THEN 190
150  IF  RND (1) <  = A / C THE
N  PRINT A$(J): GOTO 170
160 C = C - 1: GOTO 180
170 A = A - 1:C = C - 1
```

```
50 PRINT TAB(9) "AMOSTRAGEM RAN
DOMICA":PRINT:PRINT TAB(9) "DE
BUSCA INDIVIDUAL":PRINT
120 A=N:C=10
130 FOR J=1 TO 10
140 IF A=0 THEN 190
150 IF RND(1)<=A/C THEN PRINT A
```

```
$(J):GOTO 170
160 C=C-1:GOTO 180
170 A=A-1:C=C-1
```

Se você decidir selecionar três itens da lista, o primeiro deles, Bonn, só será escolhido se a função **RND** (linha 150) resultar num número menor que 3/10. Copenhague será o próximo item a ser considerado. Se Bonn já tiver entrado para a amostragem, o item Copenhague terá a chance de ser escolhido se a função **RND** gerar um valor menor que 2/9. Se Bonn não foi selecionado, a chance de Copenhague aumenta para 3/9. As linhas 160 e 170 atualizam essas probabilidades para cada elemento.

Se você comparar as amostragens obtidas por este programa com as do programa anterior, observará que, aqui, elas aparecem na ordem em que estão no programa.

A primeira vista, pode-se supor que, com uma lista de dez itens, o número de amostragens possíveis é pequeno. Na verdade, porém, é bem grande: são 120 amostras diferentes com três itens, e 252 com cinco itens.

MICRO DICAS

FAZENDO PREVISÕES

As técnicas de simulação descritas neste artigo apresentam muitos dos elementos necessários para que você próprio desenvolva um método para prever os resultados de um jogo ou para fazer uma pesquisa. Não será difícil modificar os programas ou usar parte deles, adaptando-os às suas finalidades.

Você pode, por exemplo, planejar e montar um programa que contenha o resultado de vinte partidas e, então, selecionar treze delas e comparar suas previsões com os resultados reais. Para isso, seria preciso utilizar tanto o programa da loteria esportiva quanto o da amostragem, com algumas modificações e linhas extras destinadas a ligá-los.

Se você quiser treinar um pouco mais, experimente uma outra combinação dos dois programas, fazendo com que o computador não só simule os resultados da loteria esportiva, como também forneça os nomes dos times que estão jogando. Você deverá, nesse caso, incluir no programa de amostragem os nomes de vários times, dentre os quais 26 serão sorteados.

AVALANCHE: AS PEDRAS ROLAM (2)

Willie precisará ter muito cuidado agora. As pedras estão prontas para rolar morro abaixo. E, nesta segunda parte da rotina, descerão numa avalanche sobre ele.

Até agora Willie não contava com a ameaça de uma avalanche. A rotina que faz com que as pedras rolem está apenas pela metade e, provavelmente, falhará se você tentar executá-la. Esta segunda parte da rotina irá precipitar a avalanche. E Willie só estará seguro se pular fora do caminho.

A listagem que apresentamos aqui é constituída de duas seções principais. Uma delas imprime a segunda figura da pedra — para dar a impressão de que ela está rolando — e verifica se Willie foi atingido. A outra apaga a pedra se ela tiver chegado ao fim da encosta ou à superfície do mar e a recoloca no topo da encosta. Ambas as seções são chamadas pela parte da rotina fornecida no artigo anterior.

Assim que você tiver digitado e montado as linhas que se seguem, a rotina de movimentação da pedra estará pronta para funcionar. Mas lembre-se de que você deve ter o resto do jogo na memória, já que outras rotinas — como **print** — serão chamadas.

```
 10 REM org 59097
 20 REM bma ld hl,(57356)
 30 REM ld de,22528
 40 REM add hl,de
 50  REM ld a,(hl)
 60 REM cp 40
 70 REM jr nz,bnh
 80 REM ld a,2
 90 REM ld (57336),a
100 REM bnh ld hl,(57356)
110 REM ld a,42
120 REM ld bc,57128
130 REM call 58217
140 REM inc hl
150 REM call 58217
160 REM ld a,0
170 REM ld (57358),a
180 REM ret
190 REM bri ld hl,(57356)
200 REM ld bc,15616
210 REM ld a,45
220 REM call 58217
230 REM ld hl,223
240 REM ld (57356),hl
250 REM ret
```

Na última parte de *Avalanche*, você teve que analisar a cor do caractere que estava logo abaixo da pedra, para verificar se ela tinha chegado à água ou se estava rolando. Agora, será preciso checar a cor da posição onde a pedra será impressa, para saber se ela conseguirá atingir Willie.

A posição de impressão da pedra é transferida dos endereços 57356 e 57357 para o par HL. Não se esqueça de que a variável de posição da pedra, que está nesses endereços, foi decrementada no fim da primeira parte desta rotina. Assim, quando a rotina é novamente chamada, a variável está apontando para a posição à esquerda da pedra. Saltando para essa seção da rotina, o programa imprime a segunda figura da pedra no local indicado — uma posição à esquerda da anterior —, para dar a impressão de que está em movimento.

O par DE é carregado com 22528, valor adicionado em seguida ao conteúdo de HL. O resultado, que aponta para a cor da posição onde a pedra será impressa, permanece no par HL.

Utilizamos o apontador em HL para obter a cor da posição e carregá-la no acumulador por meio de endereçamento direto. Essa cor é comparada com 40, que corresponde ao código de azul sobre fundo azul ciano, a cor de Willie contra o céu.

Se a cor nessa posição não é 40, a pedra não atingirá Willie, e a instrução **jr nz,bnh** faz o processador pular as próximas instruções.

WILLIE É ATINGIDO

Se Willie for mortalmente atingido pela pedra, a variável no endereço 57336 deve ser igualada a 2. Para isso, colocamos 2 no acumulador e carregamos seu conteúdo em 57336.

A PEDRA ESTÁ ROLANDO

Inevitavelmente, a pedra irá rolar uma posição à esquerda. Se Willie não estiver pelo caminho, nem se preocupe. Caso contrário, você pode ter a certeza de que ele foi esmagado.

O par HL é carregado com a nova posição da pedra na tela. O acumulador

A é carregado com 42 — vermelho sobre fundo ciano — e o par BC, com 57128, que corresponde ao endereço inicial dos dados para a segunda figura da pedra, que ocupa duas posições na tela. A rotina **print** é chamada e imprime a primeira metade da pedra (a porção esquerda). O par HL é incrementado — o que faz o apontador se mover uma posição para a direita. A rotina **print** é novamente chamada, imprimindo a segunda metade da pedra.

Como você verá, a segunda pedra se

desloca meio caractere de cada vez, pois ocupa duas posições na tela. Essa estrutura de duas metades faz com que o movimento seja muito mais suave e contínuo, acentuando a impressão de que a pedra está rolando.

Note que, movendo a pedra um caractere para a esquerda (como fizemos na primeira parte da rotina) e, depois, meio caractere, mantemos um avanço tão suave na posição em 57356 e 57357 que não há necessidade de verificar se a pedra ainda está no chão.

Agora, basta que igualemos a variável do tipo de pedra a 0, para que, quando a rotina for chamada de novo, o processador execute a outra parte. Para isso, colocamos 0 no acumulador e carregamos o conteúdo de A em 57358.

RECOMEÇA A AVALANCHE

A rotina **bri** é chamada pela primeira parte da rotina de movimentação sempre que a pedra chega ao fim da encosta ou atunda nas águas do mar. A função de **bri** consiste em apagar a pedra da sua posição atual e reajustá-la para o topo da encosta.

A operação de apagar a pedra é feita como de costume. Sua posição é transferida de 57356 e 57357 para HL; o par BC é carregado com o endereço inicial dos dados (que estão na ROM) para um espaço vazio e o acumulador A, com o valor 45, que corresponde ao código de ciano sobre fundo ciano. Chamada em seguida, a rotina **print** imprime um caractere cor do céu sobre a pedra, fazendo-a desaparecer da tela.

O par HL é carregado com 223, posição da pedra na tela quando ela se encontra no topo da montanha. Esse valor é colocado de volta nos endereços 57356 e 57357, para que a rotina de movimentação da pedra comece nessa posição, quando for novamente chamada.

T

A rotina aqui apresentada tem duas seções principais. Ambas são chamadas pela parte do programa publicada no artigo anterior da série *Avalanche*. A primeira delas, que começa no rótulo **BOK**, seleciona e imprime na tela um dos dois padrões de pedra. Em seguida, troca o valor da variável que controla esses padrões, para que a outra figura seja impressa na próxima vez.

A segunda seção, que começa no rótulo **BRI**, apaga a pedra se ela tiver chegado ao fim da encosta ou à água, recolocando-a no topo do morro.

Assim que tiver digitado e montado as linhas que se seguem, a rotina de movimentação da pedra estará pronta para funcionar. Mas lembre-se de que você deve ter o resto do jogo na memória, já que outras rotinas, como **CHARPR**, por exemplo, serão chamadas.

```
10   ORG 19853
20   BOK LDA 18260
30   BEQ BMN
40   LDX 18253
50   LDU #18038
60   JSR CHARPR
70   CLR 18260
80   RTS
90   BMN LDX 18253
```

```
100   LDU #18014
110   JSR CHARPR
120   LDA #1
130   STA 18260
140   RTS
150   BRI LDX 18253
160   LDU #1536
170   JSR CHARPR
180   LDX #3070
190   STX 18253
200   RTS
210   CHARPR EQU 19402
```

Existem, na memória, dois padrões diferentes da pedra, que são impressos na tela alternadamente, dando a impressão de que a figura está rolando.

Para a criação desse efeito, o processador precisa saber qual das duas figuras foi impressa na última vez. Obtém tal informação consultando uma baliza no endereço 18260. O conteúdo dessa posição é carregado no acumulador. Se for 0, a instrução **BEQ BMI** salta para a rotina que imprime uma das figuras da pedra. Se não for, o programa continua e imprime a outra.

A PEDRA ROLA

Caso o programa continue, X é carregado com o conteúdo da posição de memória 18253. Esse endereço armazena a posição onde a pedra irá ser impressa. O registrador U é carregado com o número 18038, endereço inicial de uma das figuras da pedra.

O processador salta para a rotina **CHARPR**, que imprime os últimos oito bytes da pilha do usuário na posição de tela que está sendo apontada pelo conteúdo do registrador X.

A instrução **CLR 18260** altera a baliza no endereço 18260. O processador a executa quando a baliza tem valor 1. **CLR 18260** apaga esse valor, colocando 0 em seu lugar. Assim, na próxima vez que a rotina da pedra for chamada, o processador se encarregará de executar a outra parte e imprimir a segunda figura da pedra. Depois, a baliza é zerada e o processador retorna.

A parte seguinte dessa rotina simplesmente imprime a outra figura da pedra. O registro é carregado com o mesmo valor, mas o apontador da pilha do usuário, U, é carregado com o endereço inicial na memória da outra figura da pedra, 18104. Em seguida, a rotina **CHARPR** é chamada e realiza a impressão na tela. O processador executou essa parte da rotina porque a baliza em 18260 tinha o valor 0. Conseqüentemente, 1 é carregado no acumulador e armazenado em 18260. A troca da baliza faz com que, na próxima vez, a outra figura da pedra seja impressa.

A rotina **BRI** é chamada quando a pedra chegou ao fim da encosta ou atingiu a água. Sua função consiste em apagar a pedra e inicializar sua posição para o topo da montanha. A instrução

LDX 18253 carrega no registrador X a posição atual da pedra na tela; U é carregado com o endereço inicial de um caractere de céu na memória. A rotina **CHARPR** imprime, então, um bloco de céu sobre a pedra, apagando-a.

O registrador X é carregado com 3070, posição inicial da pedra na tela quando ela se encontra no topo da montanha. Esse valor é armazenado em 18253, a variável que carrega a posição em que a pedra será impressa na próxima vez.

Para testar a rotina de movimentação da pedra, execute estas linhas em BASIC com o resto do programa.

```
5 POKE 30000,57
10 EXEC 19426
20 EXEC 19781
30 FOR K=1 TO 100:NEXT:GOTO 20
```

A linha 5 só será necessária se você tiver feito a gravação da rotina da música separadamente.

A rotina que apresentamos a seguir tem duas seções principais. Ambas são chamadas pela parte do programa publicada no artigo anterior da série *Avalanche*. A primeira seção imprime a segunda figura da pedra, que se alternará com a primeira, dando a impressão de que a pedra está rolando. Além disso, verifica também se nosso personagem foi atingido pela pedra.

A segunda seção apaga a pedra caso ela tenha chegado ao final da encosta ou à superfície do mar e inicializa sua posição no topo da montanha.

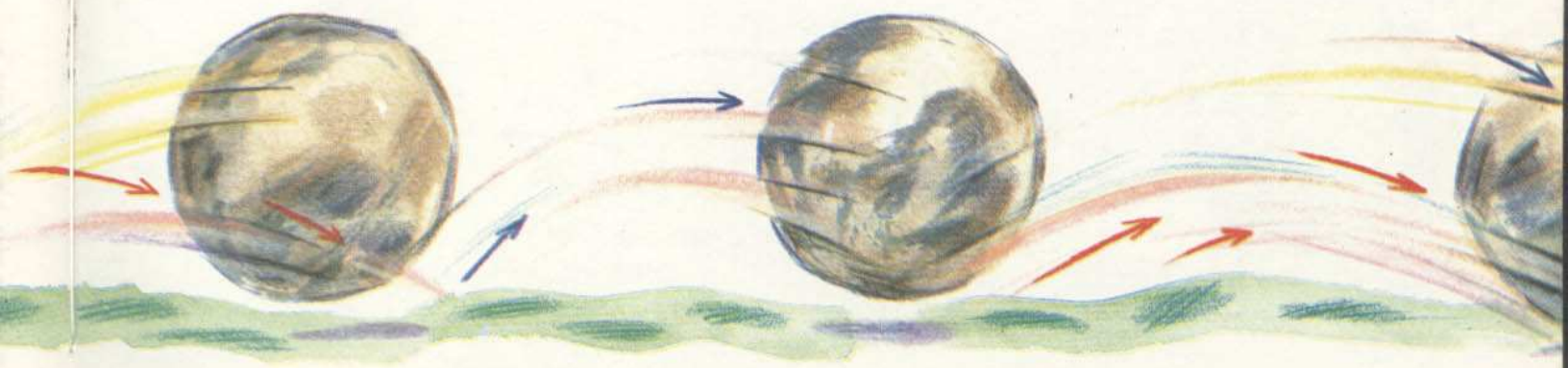
Depois de digitar e montar as próximas linhas, a rotina de movimentação da pedra estará completa e pronta para funcionar. Mas lembre-se de que o resto do jogo precisa estar na memória, pois as tabelas de padrões e de cores são necessárias nesta rotina. Para obter o efeito desejado, é preciso, também, que a montanha esteja na tela.

```
10    org 54530
20    ld hl,(62407)
30    ld de,(-5200)
40    add hl,de
50    push hl
60    call 74
70    cp 1
80    jr z,bn
90    cp 5
100    jr z,bn
110    cp 7
120    jr z,bn
130    cp 11
140    jr nz,bu
150    bn ld a,2
160    ld (-5201),a
170    bu pop hl
180    ld a,17
190    push hl
200    call 77
210    pop hl
220    inc hl
230    ld a,19
240    call 77
250    ld a,0
260    ld (-5195),a
270    ret
280    org -10953
290    ld hl,(62407)
300    ld de,(-5200)
310    add hl,de
320    ld a,255
330    call 77
340    ld hl,255
350    ld (-5200),hl
360    ret
370    end
```

Na primeira parte da rotina de movimentação, você teve que examinar o padrão que estava logo abaixo da pedra, para verificar se ela tinha afundado no mar ou se estava rolando. Identificaremos agora o padrão que está na posição onde a pedra será impressa, para saber se ela atingirá ou não Willie.

A posição de impressão da pedra, que está armazenada em −5200 e −5199, é colocada no par DE. Lembre-se de que a variável de posição da pedra, que estava nesses endereços, foi decrementada no fim da primeira parte desta rotina. Assim, quando a rotina é novamente chamada, a variável está apontando para a posição à esquerda da pedra. Ao executar essa parte da rotina, o processador imprime a segunda figura da pedra, dando a impressão de que ela está rolando.

O par HL é carregado com o endereço inicial da Tabela de Nomes da VRAM (TN). A posição que está em DE é somada ao endereço em HL, que passa a conter efetivamente o endereço equiva-



MICRO DICAS

LEITURA E ESCRITA NA VRAM DO MSX

A VRAM é tratada pelo MSX como se fosse um periférico. Portanto, são necessários alguns artifícios para colocarmos dados nessa memória auxiliar, especialmente em linguagem de máquina, quando não dispomos dos comandos **VPEEK** e **VPOKE**.

Em nosso jogo, usamos cinco rotinas para escrever e ler na VRAM. Seus endereços são: 74, 77, 86, 89 e 92. Todas empregam os comandos **OUT** e **IN** para escrever e ler através da porta 152. As rotinas dos endereços 80 e 83 controlam o destino do byte enviado por essa porta.

lente a essa posição na tela. Em seguida, o valor de HL é guardado na pilha, sendo utilizado na impressão da pedra.

A rotina 74 da ROM é chamada. Ela coloca no acumulador o valor da posição da VRAM apontada por HL, ou seja, realiza a leitura da VRAM. O acumulador passa o conter o código do padrão que está à esquerda da pedra. Esse código é comparado com 1, 5, 7 e 11, valores que correspondem aos diferentes desenhos das pernas de Willie. Caso o valor lido não seja nenhum desses, a instrução **jr nz,bu** faz o processador pular as próximas instruções.

WILLIE É ATINGIDO

Se o pobre Willie tiver sido mortalmente atingido pela pedra, a variável no endereço −5201 deve ser ajustada com o valor 2. Para isso, colocamos 2 no acumulador e carregamos o conteúdo deste em −5201.

A PEDRA ROLA

Inevitavelmente a pedra irá rolar uma posição à esquerda. Se Willie não estiver nessa posição, não se preocupe. Caso contrário, podemos ter a certeza de que ele foi esmagado.

A posição na Tabela de Nomes da VRAM é recuperada da pilha, voltando para o par de registros HL. O acumulador A é carregado com o código do primeiro padrão da figura da pedra (a metade esquerda será impressa primeiro). O valor de HL é novamente carregado na pilha, pois será usado na impressão da outra metade. A rotina 77 da ROM é chamada. Ela coloca o valor contido no acumulador na posição na VRAM apontada por HL. Já a utilizamos várias vezes na série *Avalanche*.

O endereço da VRAM é recuperado da pilha para HL, onde é incrementado. Esse par de registros passa, então, a apontar para uma posição à direita na tela, onde será impressa a metade que está faltando. O acumulador é carregado com 19, o código dessa metade, e a rotina 77 é chamada outra vez.

A segunda figura da pedra ocupa duas posições na tela, movendo-se meio caractere de cada vez. Essa estrutura de duas metades faz com que o movimento se realize de modo muito mais suave e contínuo, acentuando a impressão de que a pedra está rolando.

O deslocamento da posição da pedra, um caractere para a esquerda (como ocorreu na primeira parte da rotina) e, depois, meio caractere, mantém uma variação tão suave na posição em −5200 e −5199, que não há necessidade de verificar se a pedra está no chão.

Agora, falta apenas um detalhe: ajustar a variável do tipo de pedra a 0, para que, quando a rotina for novamente chamada, o processador execute a outra parte. Para isso, colocamos 0 no acumulador e carregamos seu conteúdo no endereço −5195.

P&R

Por que não usamos sprites para representar as pedras no MSX?

Os sprites poderiam, de fato, apresentar algumas vantagens sobre os blocos gráficos. Com eles, obteríamos os movimentos mais suaves e, também, não precisaríamos apagar a última posição da figura já que isso é feito automaticamente.

Mas não foi sem razão que optamos pelos blocos gráficos. Primeiro, só é permitida uma cor no sprite. Além disso, a detecção da colisão de um sprite com outras estruturas complicaria muito o programa. Por fim, não poderíamos ter, como requer o jogo, mais do que quatro figuras simultaneamente na mesma linha.

RECOMEÇA A AVALANCHE

A rotina **mo** é chamada pela primeira parte da rotina de movimentação sempre que a pedra chega ao fim da encosta ou atinge o mar. Ela apaga a pedra da sua posição atual e reajusta essa posição para o topo da montanha.

Para apagar a pedra, o endereço inicial da TN da VRAM é colocado no par de registros HL. A esse endereço somase a posição atual da pedra, que está em DE. O valor 255, que corresponde ao código do padrão de céu, é colocado no acumulador. A rotina 77 é chamada e imprime o padrão de céu na posição que a pedra ocupava.

O par HL é carregado com 255, a posição da pedra no topo da encosta. Esse valor volta para −5200 e −5199. Quando a rotina de movimentação for chamada, a pedra estará nessa posição.

DISCOS RÍGIDOS

Embora muito úteis, os disquetes têm um inconveniente: a baixa capacidade de armazenamento. Se você precisa de maior espaço de memória e velocidade de acesso, o disco rígido é a solução.

Em artigos anteriores, discutimos as características e as vantagens dos discos flexíveis (também chamados *floppies* ou disquetes) para os usuários de microcomputadores. Esses periféricos são formidáveis, em termos de capacidade de armazenamento, facilidade de uso e velocidade de acesso, quando comparados com outras formas de gravação magnética de informação.

Entretanto, também apresentam desvantagens. A principal é a incapacidade de atender a demandas de armazenamento maiores do que as habituais.

Os disquetes de face simples (para as linhas TRS-80, TRS-Color, Apple e TK-2000) têm capacidade em torno dos 160-180 Kbytes. Pode parecer muito, mas é suficiente apenas para cerca de noventa páginas de texto, ou alguns programas e arquivos de dados pequenos. Os disquetes de dupla face têm capacidade de armazenamento de cerca de 320-360 Kbytes — o que também não é muito, considerando-se a espantosa rapidez com que um usuário médio enche até centenas de disquetes.

Não seria interessante ter todos os programas e arquivos de dados em um único disco? Para isso, existe uma solução: o disco rígido. Embora ainda seja um periférico muito caro (sobretudo no Brasil), seu preço tende a se tornar acessível. Não há exagero em afirmar que, mais cedo ou mais tarde, todos os micros pessoais serão vendidos com uma unidade embutida de disco rígido, como já ocorre com os micros profissionais da linha PC-XT.

O QUE É UM DISCO RÍGIDO

O disco rígido (*hard disk*, em inglês) é feito de metal, e não de plástico flexível, como o disquete. Por essa razão, apresenta mais estabilidade térmica e estrutural, o que lhe permite maiores velocidades de rotação (e acesso) e maior densidade de gravação. Essas características resultam numa grande capacidade de armazenamento.

A desvantagem do disco rígido é que, em geral, ele não pode ser removido e trocado facilmente por outro, como o disquete, pois é fixo dentro da unidade acionadora. Existem unidades de disco rígido que são totalmente intercambiáveis — inclusive a cabeça de gravação e leitura.

Nos discos rígidos mais utilizados para micros, a cabeça de gravação e leitura nunca entra em contato com a superfície do disco, como acontece com o disquete. Ela "sobrevoa" a superfície a uma distância muito pequena (a alguns milésimos de milímetro); por isso, o desgaste da superfície ferromagnética é praticamente nulo.

Esse sistema garante maior durabilidade à unidade, mas requer que ela seja isolada do exterior, por meio de vácuo ou fluxo forçado de ar. A partícula mais ínfima de poeira ou fumaça que se introduzir entre a cabeça e o disco pode danificá-lo.

Essa tecnologia é conhecida como *Winchester*, denominação que se costuma estender aos próprios discos rígidos. Como ela envolve dispositivos mecânicos e eletrônicos complexos e delicados, os custos dos discos rígidos tornam-se bem mais caros.

CAPACIDADE E VELOCIDADE

A capacidade de armazenamento dos discos Winchester é espantosa. Os de menor capacidade têm por volta de 5 Mbytes de espaço (5 milhões de caracteres), o que equivale a cerca de trinta disquetes de face simples!

São cada vez mais populares os discos rígidos de 10, 15 e 20 Mbytes, embora as maiores capacidades possam ser usadas apenas por micros de 16 e 32 bits. Mas já existem discos de boa capacidade para o Apple e o TRS-80.

A velocidade do disco rígido é cerca de vinte a trinta vezes maior que a dos disquetes, dependendo do modelo. Isso faz uma enorme diferença no momento de carregar um programa extenso na memória do micro.

COMO CONECTAR UM DISCO RÍGIDO

Desde que existam modelos de disco rígido disponíveis para o seu microcomputador, você não terá dificuldades em utilizar este periférico.

Em primeiro lugar, será necessário uma interface controladora — uma caixinha ou placa, que pode ser conectada ao computador (internamente, por exemplo, nos micros da linha Apple), facultando-lhe o controle e intercâmbio de dados com a unidade de disco.

Muitos micros de oito bits têm placas de controle para disquetes que já incluem o controlador de discos rígidos. Em alguns casos, porém, é preciso adquirir uma interface própria.

A unidade de disco rígido é apresentada normalmente em duas versões: em gabinetes separados do console do computador, e em "gavetas" que podem ser inseridas no local destinado a uma unidade acionadora de disquetes. Em alguns computadores (como os PC) é possível instalar a unidade Winchester em espaço reservado no gabinete da UCP, sem tomar o lugar de um disquete.

Não convém dispor apenas do disco rígido: é melhor ter também uma ou duas unidades de disquetes — só assim poderemos obter a cópia cautelar (*back-up*) do disco rígido e a transferência de programas e dados.

O disco rígido tem de fato a desvantagem de requerer freqüentemente cópias de seu conteúdo em disquete, pois, se ocorrer um defeito, a perda é total. Por isso, muitos usuários adquirem junto com o Winchester uma unidade de fita de back-up (chamada *streamer*), que possibilita a execução dessa cópia de uma vez só, e em poucos minutos.

Resta considerar a fonte de alimentação. O disco rígido gasta mais energia do que uma unidade de disquetes, exigindo, quase sempre, uma fonte de alimentação separada da UCP. Alguns fabricantes vendem a unidade de disco rígido com fonte própria de alimentação, o qual é muito conveniente, sobretudo se a unidade não é interna.

UMA PLANILHA ELETRÔNICA (2)

Descubra para onde vai o seu dinheiro ou planeje o futuro de seus negócios utilizando esta prática planilha. Adicione mais uma parte ao programa iniciado no artigo anterior.

Quando uma planilha começa a ser planejada e está como uma folha em branco, muitas vezes ainda não sabemos exatamente de que maneira iremos aproveitá-la. Os exemplos dados no artigo anterior e as sugestões que aqui apresentamos vão ajudá-lo a definir uma planilha que seja realmente útil. O programa permite a montagem de diversas planilhas, que você poderá gravar e recarregar a qualquer momento.

Uma planilha para registrar e planejar suas despesas domésticas, onde as entradas aparecem sob títulos como aluguel, transporte, saúde, consertos etc. constitui uma boa opção. Mas, se você tem que fazer um número muito grande de consertos ou reformas na casa, por exemplo, pode ser interessante montar uma planilha só para eles. Nesse caso, separe os diferentes tipos de reforma — estofamento dos móveis, decoração, troca das telhas e calhas, colocação de grades nas janelas —, especificando as quantias gastas mensalmente ou trimestralmente com cada um. O programa se encarregará de fornecer os valores totais para cada categoria, assim como seu peso relativo no conjunto dos gastos com reformas.

Uma outra folha pode incluir as despesas da família com itens como alimentação, vestuário, educação, saúde, transporte e lazer. Utilize-a para listar esses gastos por semanas, meses ou por pessoas da família.

Mas lembre-se de que pode recorrer à planilha para manipular qualquer tipo de informação que necessite de uma organização lógica. Uma planilha para sócios de um clube pode conter, por exemplo, os nomes, telefones e mensalidades pagas, assim como os comparecimentos a reuniões.

Qualquer que seja sua escolha, a planilha se revelará um excelente instrumento de controle. Se você consultar o artigo da página 201, verá que ela nada mais é que uma sofisticada matriz bidimensional. Permite um controle maior dos dados porque comporta o uso de fórmulas, o que torna possível a obtenção imediata dos resultados.

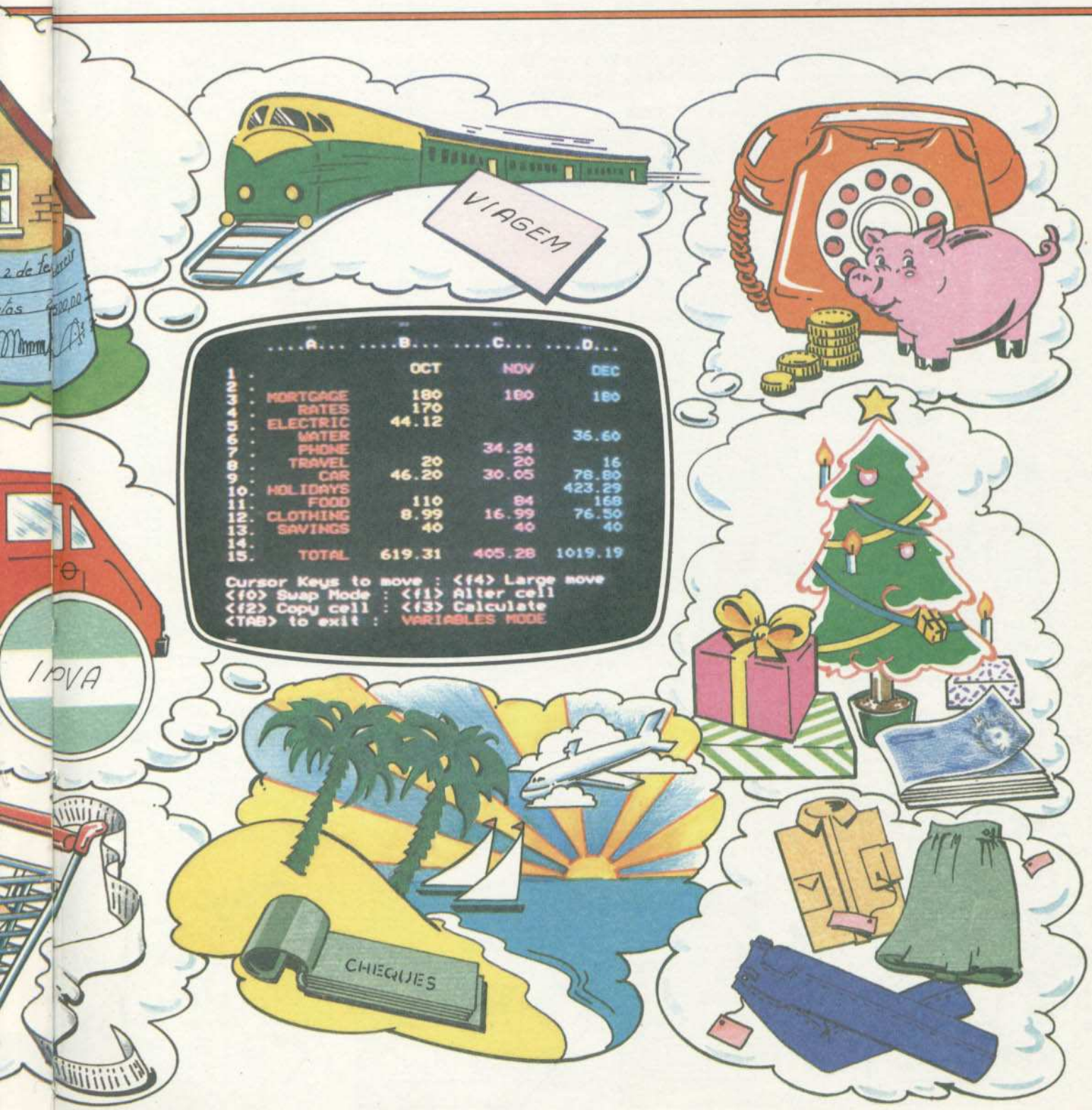
Aplicações financeiras constituem o tipo mais freqüente de uso. Mas as variações, nessa área, também são ilimitadas. Planilhas podem conter detalhes de pedidos, descrições de itens, evolução nos custos, descontos etc. São úteis, também, na elaboração de folhas de pagamento, fornecendo listagens dos nomes dos empregados e calculando horas trabalhadas, salários e adiantamentos. Sua cooperação é, ainda, valiosa no controle de estoque e contas, em geral; no planejamento de orçamentos de firmas e em muitos outras tarefas ligadas ao setor empresarial.

DIGITE O PROGRAMA

A parte do programa aqui listada deve ser adicionada à que apresentamos no artigo anterior. As linhas restantes serão dadas no último artigo da série, que conterá instruções detalhadas sobre o uso do programa. Assim, carregue a listagem anterior, digite esta e grave o programa para, depois, acrescentar a terceira parte.

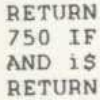
```
420 FOR a=fc TO c: FOR b=fr TO
tr
430 IF z$(3,2)="C" THEN  LET v
(2)=v(2)+1: LET v(4)=v(4)+(v(3
)<>26)
440 IF z$(3,2)="R" THEN  LET v
(3)=v(3)+(v(3)<>26): LET v(1)=
v(1)+1
450 IF v(1)<25 AND v(2)<31 AND
v(3)=26 THEN  GOTO 470
460 IF v(1)>24 OR v(2)>30 OR v
(3)>24 OR v(4)>30 THEN  GOTO
 570
470 IF v(1)<1 OR v(2)<1 OR v(3
)<1 OR v(4)<1 THEN  GOTO 570
480 LET a$=CHR$ (v(1)+64)+STR$
v(2)+CHR$ (v(3)+64)+STR$ v(4)+
o$
490 LET c=LEN a$: IF c>8 THEN
RETURN : RESTORE 1630: FOR q=1
TO 11: LET f=0: READ m$: FOR w
=1 TO c
500 IF m$(w)="A" THEN  GOSUB
1650: IF f THEN  GOTO 560
510 IF m$(w)="N" THEN  GOSUB
1670: IF f THEN  GOTO 560
520 IF m$(w)="Z" THEN  GOSUB
1710: IF f THEN  GOTO 560
530 IF m$(w)="O" THEN  GOSUB
1690: IF f THEN  GOTO 560
```

- PLANEJAMENTO DA PLANILHA
- USOS GERAIS
- ORÇAMENTO FAMILIAR
- DESPESAS DOMÉSTICAS
- CONTROLE DE
- UMA ASSOCIAÇÃO
- USOS FINANCEIROS
- SALÁRIOS DE PESSOAL
- ESTOCAGEM
- DIGITE O PROGRAMA

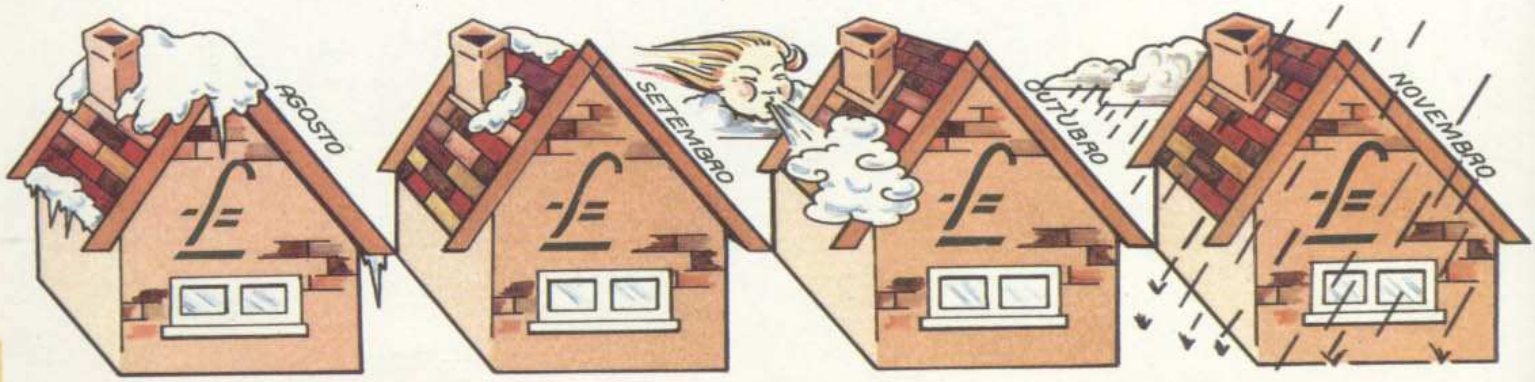



```
     ....A....  ....B....  ....C....  ....D....
1.
2.                  OCT        NOV        DEC
3.      MORTGAGE    180        180        180
4.         RATES    170
5.      ELECTRIC  44.12
6.         WATER                          36.60
7.         PHONE               34.24
8.        TRAVEL      20         20         16
9.           CAR   46.20      30.05      78.80
10.     HOLIDAYS                         423.29
11.         FOOD     110         84        168
12.     CLOTHING    8.99      16.99      76.50
13.      SAVINGS      40         40         40
14.
15.        TOTAL  619.31     405.28    1019.19

Cursor Keys to move : <f4> Large move
<f0> Swap Mode : <f1> Alter cell
<f2> Copy cell : <f3> Calculate
<TAB> to exit : VARIABLES MODE
```

```
540 NEXT w: LET z=q: GOSUB
1140: IF NOT f THEN  LET s$="
      ": FOR w=1 TO c: LET s$
(w)=a$(w): NEXT w: LET d$(b,a,
9 TO 16)=s$: LET d$(b,a,18)=
CHR$ z: LET d$(b,a,17)="1":
NEXT b: NEXT a: RETURN
550 GOTO 570
560 NEXT q
570 RETURN
580 LET e=c: LET a$="    "
590 PRINT #1;AT 0,x; BRIGHT 1;
" "
600 PAUSE 0: LET i=CODE INKEY$
610 IF i>88 THEN  GOTO 600
620 IF i=13 THEN  GOTO 650
630 IF i=12 THEN  LET a$(4-e)=
" ": LET e=e+1: LET x=x-1
640 LET a$(4-e)=CHR$ i: PRINT
#1;AT 0,x;CHR$ i: LET x=x+1:
LET e=e-1: IF e>0 THEN  PAUSE
10: GOTO 590
650 IF e>1 AND (d=1 OR d=4 OR
d=5) THEN  GOTO 590
655 IF e>0 AND (d=2 OR d=3)
THEN  GOTO 590
660 PAUSE 10: PRINT #1;AT 0,0;
"
  ": RETURN
670 LET i$=""
680 FOR z=1 TO 3
690 LET i$=i$+(a$(z) AND a$(z)
<>" ")
700 NEXT z
710 IF LEN i$=3 THEN  IF i$(1)
<"A" OR i$(1)>"X" OR i$(2)<"0"
OR i$(2)>"9" OR i$(3)<"0" OR i
$(3)>"9" THEN  LET f=1: RETURN

720 IF LEN i$=3 THEN  IF VAL
i$(2 TO 3)=0 OR VAL i$(2 TO 3)
>30 THEN  LET f=1: RETURN
730 IF LEN u$=2 THEN  IF i$(1)
<"A" OR i$(1)>"X" OR i$(2)<"1"
OR i$(2)>"9" THEN  LET f=1:
RETURN
740 IF d=2 THEN  IF i$(1)<>"A"
AND i$(1)<>"R" THEN  LET f=1:
RETURN
750 IF d=3 THEN  IF i$(1)<>"C"
AND i$(1)<>"R" THEN  LET f=1:
RETURN
760 LET z$(d,2 TO )=i$: LET z$
(d,1)=CHR$ (LEN i$+48): LET f=
0: RETURN
770 LET fc=(CODE z$(4,2))-64:
LET tc=(CODE z$(5,2))-64: LET
fr=VAL z$(4,3 TO (1+VAL z$(4,1
))): LET tr=VAL z$(5,3 TO (1+
VAL z$(5,1)))
780 IF z$(3,2)="C" THEN  IF fc
<>tc OR fr>tr THEN  LET f=1:
RETURN
790 IF z$(3,2)="R" THEN  IF fc
>tc OR fr<>tr THEN  LET f=1:
RETURN
800 LET f=0: RETURN
810 FOR y=1 TO 30: FOR x=1 TO
24: LET os=0
820 IF d$(y,x,17)="1" THEN
LET z=CODE d$(y,x,): GOSUB 880
: GOSUB 1010: LET st=1: LET a$
=STR$ t: LET os=LEN a$: IF t>
99999.99 THEN  LET f$(y,x,1)=
"5"
830 IF os>8 THEN  LET st=os-7
840 IF os=0 THEN  GOTO 860
850 LET s$="          ": FOR u=s
t TO os: LET s$(u-st+1)=a$(u):
NEXT u: GOSUB 1410: LET d$(y,x
, TO 8)=s$
860 LET i=IN 32766: IF i=252
THEN  PRINT #1;AT 0,0; PAPER 2
; INK 7;"CALCULO ABANDONADO":
RETURN
870 NEXT x: NEXT y: RETURN
880 LET s$=d$(y,x,9 TO 16)
890 IF z=1 THEN  LET v(1)=(
CODE s$(1))-64: LET v(2)=VAL s
$(2): LET v(3)=(CODE s$(3))-64
: LET v(4)=VAL s$(4): LET o$=s
$(5): RETURN
900 IF z=2 THEN  LET v(1)=(
CODE s$(1))-64: LET v(2)=VAL s
$(2 TO 3): LET v(3)=(CODE s$(4
)-64): LET v(4)=VAL s$(5): LET
 o$=s$(6): RETURN
```



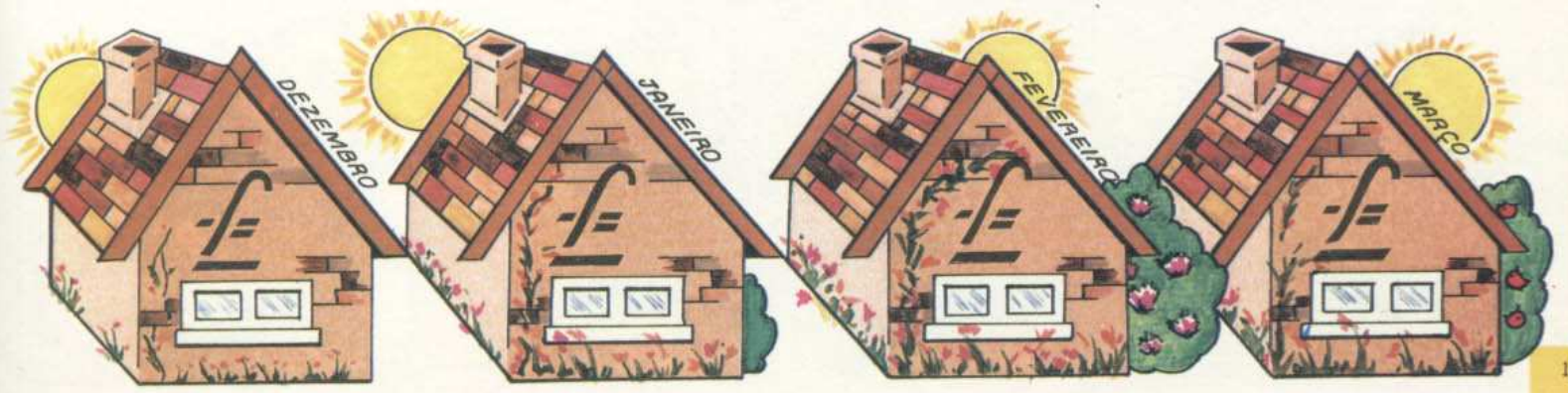
```
660 A$=D$(I,J):B$=MID$(A$,2)
670 AT=ASC(A$)
680 IF AT=128 THEN PRINT STRING
$(7,32);:GOTO 720
690 IF AT=129 OR AT=130 THEN PR
INT USING "%      %";B$;:GOTO 72
0
700 FOR U=1 TO LEN(B$):IF MID$(
B$,U,1)<>CHR$(32) THEN PRINT MI
D$(B$,U,1);
710 NEXT U
720 RETURN
730 C1=ASC(Z$)-64:C2=VAL(MID$(Z
$,2)):V=D(C1,C2):RETURN
740 PRINT @448,"TRABALHANDO"
750 FOR J=1 TO RX
760 FOR I=1 TO CX
770 D(I,J)=0:IF ASC(D$(I,J))=12
9 THEN D(I,J)=VAL(MID$(D$(I,J),
2))
780 NEXT I,J
790 FOR J=1 TO RX
800 FOR I=1 TO CX
810 PRINT @448,"TRABALHANDO NA
CELULA ";CHR$(I+64);MID$(STR$(J
),2)
820 IF ASC(D$(I,J))<>131 THEN 1
130
830 A$=MID$(D$(I,J),2)
840 O$=MID$(A$,7,1)
850 IF O$="&" THEN 1050
860 IF O$="$" THEN 1090
870 Z$=LEFT$(A$,3)
880 GOSUB 730
890 V1=V:Z$=MID$(A$,4,3):GOSUB
730:V2=V
900 DP=VAL(RIGHT$(A$,1))
910 ON INSTR (1,OP$,O$) GOSUB 1
000,1010,1020,1030,1040
920 OV=0:IF DP=0 THEN PU$="####
###":MP=7:GOTO 950
930 PU$=STRING$(7-(DP+1),"#")+"
."+STRING$(DP,"#"):MP=7-(DP+1)
940 IF LEN(PU$)>7 THEN RV$="  <
OV>":OV=1
950 D(I,J)=RV
960 IF RV<0 THEN MP=MP-1
970 ML=LEN(MID$(STR$(INT(RV+.5)
),2))
980 IF ML>MP THEN RV$="  <OV>":
OV=1
990 GOTO 1160
1000 RV=V1+V1:RETURN
1010 RV=V1-V2:RETURN
1020 RV=V1*V2:RETURN
1030 IF V2=0 THEN RV=0:RETURN E
LSE RV=V1/V2:RETURN
1040 RV=V1*V2/100:RETURN
1050 P1=ASC(A$)-64:P2=ASC(MID$(
A$,4,1))-64:C2=VAL(MID$(A$,2,2)
):RV=0
1060 FOR C1=P1 TO P2
1070 RV=RV+D(C1,C2):NEXT
1080 DP=VAL(RIGHT$(A$,1)):GOTO
920
1090 P1=VAL(MID$(A$,2,2)):P2=VA
L(MID$(A$,5,2)):C1=ASC(A$)-64:R
V=0
1100 FOR C2=P1 TO P2
1110 RV=RV+D(C1,C2):NEXT
1120 DP=VAL(RIGHT$(A$,1)):GOTO
920
1130 IF ASC(D$(I,J))<>128 THEN
```



```
1150
1140 RV$=STRING$(7,32):GOTO 116
0
1150 RV$=MID$(D$(I,J),2)
1160 IF I>=CS AND I<=CS+3 AND J
>=RS AND J<=RS+11 THEN PRINT @(
J-RS)*32+35+(I-CS)*7,"";:PF=1 E
LSE PF=0
1170 IF(ASC(D$(I,J))>=128 AND A
SC(D$(I,J))<=130) OR OV=1 THEN
1200
1180 IF PF=1 THEN PRINT USING P
U$;RV;
1190 GOTO 1210
1200 IF PF=1 THEN PRINT USING"\
  \";RV$;
1210 NEXT I,J
1220 RETURN
1230 CLS:INPUT"DESEJA SALVAR ES
TA FOLHA (S/N) ";A$
1240 IF A$<>"S" THEN 1340
1250 LINE INPUT"NOME DO ARQUIVO
: ";F$
1260 OPEN "O",#-1,F$
1270 FOR J=1 TO RX
1280 FOR I=1 TO CX
1290 IF ASC(D$(I,J))=128 THEN 1
320
1300 Z$=D$(I,J):MID$(Z$,1,1)=CH
R$(ASC(MID$(Z$,1,1))-95)
1310 PRINT #-1,STR$(I),STR$(J):
PRINT #-1,Z$
1320 NEXT I,J
1330 CLOSE #-1
1340 CLS:MO=1:GOSUB 70:RETURN
1350 CLS:PRINT"DESEJA CARREGAR
UMA FOLHA DO    GRAVADOR ?":PRI
NT"(O CONTEUDO DA MEMORIA SERA
    ADICIONADO AO DA FITA)":INP
UT "S / N ";A$
1360 IF A$<>"S" THEN 1480
1370 PRINT"PRESSIONE <ENTER> PA
RA CARREGAR O PROXIMO ARQUIVO D
A FITA  OU   DIGITE NOME DO ARQ
UIVO DESEJADO":PRINT
1380 LINE INPUT"NOME DO ARQUIVO
: ";F$
1390 OPEN "I",#-1,F$
1400 IF EOF(-1) THEN 1470
1410 INPUT#-1,A$,B$:LINE INPUT#
-1,C$
1420 MID$(C$,1,1)=CHR$(ASC(MID$
(C$,1,1))+95
1430 C1=VAL(A$):C2=VAL(B$):D$(C
1,C2)=C$
1440 IF C1>CX THEN CX=C1
1450 IF C2>RX THEN RX=C2
1460 GOTO 1400
1470 CLOSE #-1
1480 CLS:CC=1:CR=1:CS=1:RS=1:MO
=1:GOSUB 70:RETURN
```

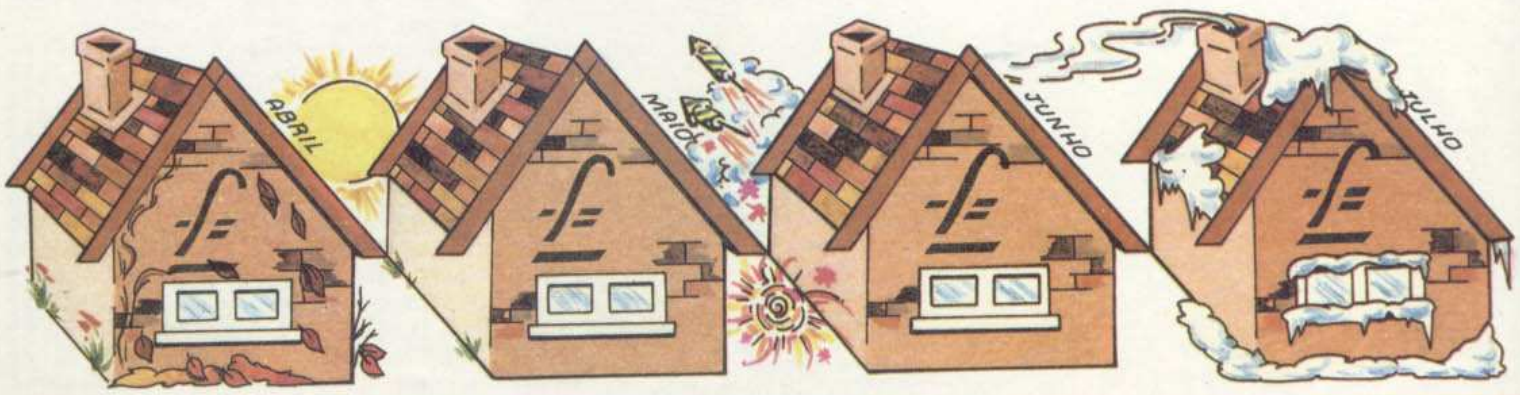


```
660 A$=D$(I,J):B$=MID$(A$,2)
670 AT=ASC(A$)
680 IF AT=128 THEN PRINTSTRING$
(7,32):GOTO 720
690 IF AT=129 OR AT=130 THEN PR
INTUSING"\      \";B$;:GOTO 720
700 FOR U=1 TO LEN(B$):IF MID$(
B$,U,1)<>CHR$(32)THEN PRINTMID$
(B$,U,1);
710 NEXTU
720 RETURN
730 C1=ASC(Z$)-64:C2=VAL(MID$(Z
$,2)):V=D(C1,C2):RETURN
740 LOCATE 0,20:PRINT"TRABALHAN
DO...";SPC(24)
750 FOR J=1 TO RX
760 FOR I=1 TO CX
770 D(I,J)=0:IF ASC(D$(I,J))=12
9 THEN D(I,J)=VAL(MID$(D$(I,J),
2))
780 NEXT I,J
790 FOR J=1 TO RX
800 FOR I=1 TO CX
810 LOCATE 0,20:PRINT"TRABALHAN
DO NA CEL ";CHR$(I+64);MID$(STR
$(J),2)
820 IF ASC(D$(I,J))<>131 THEN 1
130
830 A$=MID$(D$(I,J),2)
840 O$=MID$(A$,7,1)
850 IF O$="&" THEN 1050
860 IF O$="$" THEN 1090
870 Z$=LEFT$(A$,3)
880 GOSUB 730
890 V1=V:Z$=MID$(A$,4,3):GOSUB
730:V2=V
900 DP=VAL(RIGHT$(A$,1))
910 ON INSTR(1,OP$,O$) GOSUB 10
00, 1010, 1020, 1030, 1040
920 OV=0:IF DP=0 THEN PU$="####
###":MP=7:GOTO 950
930 PU$=STRING$(7-(DP+1),"#")+"
 "+STRING$(DP,"#"):MP=7-(DP+1)
940 IF LEN(PU$)>7 THEN RV$="  <
OV>":OV=1
950 D(I,J)=RV
960 IF RV<0 THEN MP=MP-1
970 ML=LEN(MID$(STR$(INT(RV+.5)
),2))
980 IF ML>MP THEN RV$="  <OV>":
OV=1
990 GOTO 1160
1000 RV=V1+V2:RETURN
1010 RV=V1-V2:RETURN
1020 RV=V1*V2:RETURN
1030 IF V2=0 THEN RV=0:RETURN E
LSE RV=V1/V2:RETURN
1040 RV=V1*V2/100:RETURN
1050 P1=ASC(A$)-64:P2=ASC(MID$(
A$,2,2)):RV=0
1060 FOR C1=P1 TO P2
1070 RV=RV+D(C1,C2):NEXT
1080 DP=VAL(RIGHT$(A$,1)):GOTO
920
1090 P1=VAL(MID$(A$,2,2)):P2=VA
L(MID$(A$,5,2)):C1=ASC(A$)-64:R
V=0
1100 FOR C2=P1 TO P2
1110 RV=RV+D(C1,C2):NEXT
1120 DP=VAL(RIGHT$(A$,1)):GOTO
920
1130 IF ASC(D$(I,J))<>128 THEN
1150
1140 RV$=STRING$(7,32):GOTO 116
0
1150 RV$=MID$(D$(I,J),2)
1160 IF I>=CS AND I<=CS+4 AND J
>=RS AND J<=RS+15 THEN LOCATE (
I-CS)*7+3,(J-RS)+1:PF=1 ELSE PF
=0
1170 IF (ASC(D$(I,J))>=128 AND
ASC(D$(I,J))<=130) OR OV=1 THEN
 1200
1180 IF PF=1 THEN PRINTUSINGPU$
;RV;
1190 GOTO 1210
1200 IF PF=1 THEN PRINTUSING"\
   \";RV$;
1210 NEXT I,J
1220 RETURN
1230 CLS:INPUT"QUER GRAVAR ESTA
 FOLHA? (S/N) ";A$
1240 IF A$<>"S" THEN 1340
1250 LINEINPUT "NOME DO ARQUIVO
: ";F$
1260 F$="CAS:"+F$:OPEN F$ FOR O
UTPUT AS #1
1270 FOR J=1 TO RX
1280 FOR I=1 TO CX
1290 IF ASC(D$(I,J))=128 THEN 1
320
1300 Z$=D$(I,J):MID$(Z$,1,1)=CH
R$(ASC(MID$(Z$,1,1))-90)
1310 PRINT#1,STR$(I);",";STR$(J
):PRINT#1,Z$
```



```
1320 NEXT I,J
1330 CLOSE #1
1340 CLS:MO=1:GOSUB 70:RETURN
1350 CLS:INPUT"QUER CARREGAR UM
A FOLHA DO TAPE?        (NOTE QU
E A FOLHA DA MEMÓRIA SERÁ COMBI
 NADA COM A NOVA) ";A$
1360 IF A$<>"S" THEN 1480
1380 LINEINPUT "NOME DO ARQUIVO
: ";F$
1390 F$="CAS:"+F$:OPEN F$ FOR I
NPUT AS #1
1400 IF EOF(1) THEN 1470
1410 INPUT #1,A$,B$:LINEINPUT #
1,C$
1420 MID$(C$,1,1)=CHR$(ASC(MID$
(C$,1,1))+90)
1430 C1=VAL(A$):C2=VAL(B$):D$(C
1,C2)=C$
1440 IF C1>CX THEN CX=C1
1450 IF C2>RX THEN RX=C2
1460 GOTO 1400
1470 CLOSE #1
1480 CLS:CC=1:CR=1:CS=1:RS=1:MO
=1:GOSUB70:RETURN
```

```
660 A$ = D$(I,J):B$ =  MID$ (A$
,2)
670 AU =  ASC (A$)
680  IF AU = 128 THEN  PRINT  S
PC( 7);: GOTO 720
690  IF AU = 129 OR AU = 130 TH
EN  PRINT  LEFT$ (B$,7); SPC( 7
 -  LEN ( LEFT$ (B$,7))): GOTO
720
700 UU = 0: FOR U = 1 TO  LEN (
B$): IF  MID$ (B$,U,1) <  > CH
R$ (32) THEN  PRINT  MID$ (B$,U
,1);:UU = UU + 1
710  NEXT : PRINT  SPC( 7 - UU)

720  RETURN
730 C1 =  ASC (Z$) - 64:C2 =  V
AL ( MID$ (Z$,2)):V = D(C1,C2):
 RETURN
740  VTAB 21: HTAB 1: PRINT "TR
ABALHANDO"
750  FOR J = 1 TO RX
760  FOR I = 1 TO CX
770 D(I,J) = 0: IF  ASC (D$(I,J
)) = 129 THEN D(I,J) =  VAL ( M
ID$ (D$(I,J),2))
780  NEXT : NEXT
790  FOR J = 1 TO RX
800  FOR I = 1 TO CX
810  VTAB 21: HTAB 1: PRINT "TR
ABALHANDO NA CEL "; CHR$ (I + 6
4);J
820  IF  ASC (D$(I,J)) <  > 131
 THEN 1130
830 A$ =  MID$ (D$(I,J),2)
840 O$ =  MID$ (A$,7,1)
850  IF O$ = "&" THEN 1050
860  IF O$ = "$" THEN 1090
870 Z$ =  LEFT$ (A$,3)
880  GOSUB 730
890 V1 = V:Z$ =  MID$ (A$,4,3):
 GOSUB 730:V2 = V
900 DP =  VAL ( RIGHT$ (A$,1))
905 IN = 1
910  IF O$ <  > MID$ (OP$,IN,1
) THEN IN = IN + 1: GOTO 910
915  ON IN GOSUB 1000,1010,1020
,1030,1040
920 OV = 0:MP = 7
950 D(I,J) = RV
960  IF RV < 0 THEN MP = MP - 1

970 ML =  LEN ( STR$ ( INT (RV
+ .5)))
980  IF ML > MP THEN RV$ = "  <
OV>":OV = 1
990  GOTO 1160
1000 RV = V1 + V2: RETURN
1010 RV = V1 - V2: RETURN
1020 RV = V1 * V2: RETURN
1030  IF V2 = 0 THEN RV = 0: RE
TURN
1035 RV = V1 / V2: RETURN
1040 RV = V1 * V2 / 100: RETURN

1050 P1 =  ASC (A$) - 64:P2 =
ASC ( MID$ (A$,4,1)) - 64:C2 =
 VAL ( MID$ (A$,2,2)):RV = 0
1060  FOR C1 = P1 TO P2
1070 RV = RV + D(C1,C2): NEXT
1080 DP =  VAL ( RIGHT$ (A$,1))
: GOTO 920
1090 P1 =  VAL ( MID$ (A$,2,2))
:P2 =  VAL ( MID$ (A$,5,2)):C1
=  ASC (A$) - 64:RV = 0
1100  FOR C2 = P1 TO P2
1110 RV = RV + D(C1,C2): NEXT
1120 DP =  VAL ( RIGHT$ (A$,1))
: GOTO 920
1130  IF  ASC (D$(I,J)) <  > 12
8 THEN 1150
1140 RV$ = "        "
1150 RV$ =  MID$ (D$(I,J),2)
1160  IF I >  = CS AND I <  = C
S + 4 AND J >  = RS AND J <  =
RS + 11 THEN  VTAB J - RS + 2:
HTAB (I - CS) * 7 + 4:PF = 1: G
OTO 1170
1165 PF = 0
1170  IF ( ASC (D$(I,J)) >  = 1
28 AND  ASC (D$(I,J)) <  = 130)
 OR OV = 1 THEN 1200
1180  IF PF = 1 THEN  PRINT  SP
C( 7 -  LEN ( LEFT$ ( STR$ (RV)
,7))); LEFT$ ( STR$ (RV),7);
1190  GOTO 1210
1200  IF PF = 1 THEN  PRINT RV$
;
1210  NEXT : NEXT
1220  RETURN
1230  HOME : INPUT "QUER GRAVAR
 ESTA FOLHA? (S/N) ";A$
1240  IF A$ <  > "S" THEN 1340
1250  INPUT "NOME DO ARQUIVO ";
F$
1260  PRINT D$;"OPEN ";F$: PRIN
T D$;"WRITE ";F$
1270  FOR J = 1 TO RX
1280  FOR I = 1 TO CX
1290  IF  ASC (D$(I,J)) = 128 T
HEN 1320
1300 Z$ = D$(I,J):Z$ =  CHR$ (
ASC (Z$) - 90) +  MID$ (Z$,2)
1310  PRINT  STR$ (I); CHR$ (13
); STR$ (J); CHR$ (13);Z$
1320  NEXT : NEXT
1330  PRINT D$;"CLOSE"
1340  HOME :MO = 1: GOSUB 70: R
ETURN
1350  HOME : INPUT "QUER CARREG
AR UMA FOLHA DO DISCO?        (N
OTE QUE A FOLHA DA MEMORIA SERA
 COMBI NADA COM A NOVA)     (S/N
)";A$
1360  IF A$ <  > "S" THEN 1480
1380  INPUT "NOME DO ARQUIVO ";
F$
1390  PRINT D$;"OPEN ";F$
1400  PRINT D$;"READ ";F$
1410  INPUT A$,B$,C$
1420 C$ =  CHR$ ( ASC (C$) + 90
) +  MID$ (C$,2)
1430 C1 =  VAL (A$):C2 =  VAL (
B$):D$(C1,C2) = C$
1440  IF C1 > CX THEN CX = C1
1450  IF C2 > RX THEN RX = C2
1460  GOTO 1410
1470  PRINT D$;"CLOSE"
1480  HOME :CC = 1:CR = 1:CS =
1:RS = 1:MO = 1: GOSUB 70: GOTO
 320
```



O JOGO DA SENHA

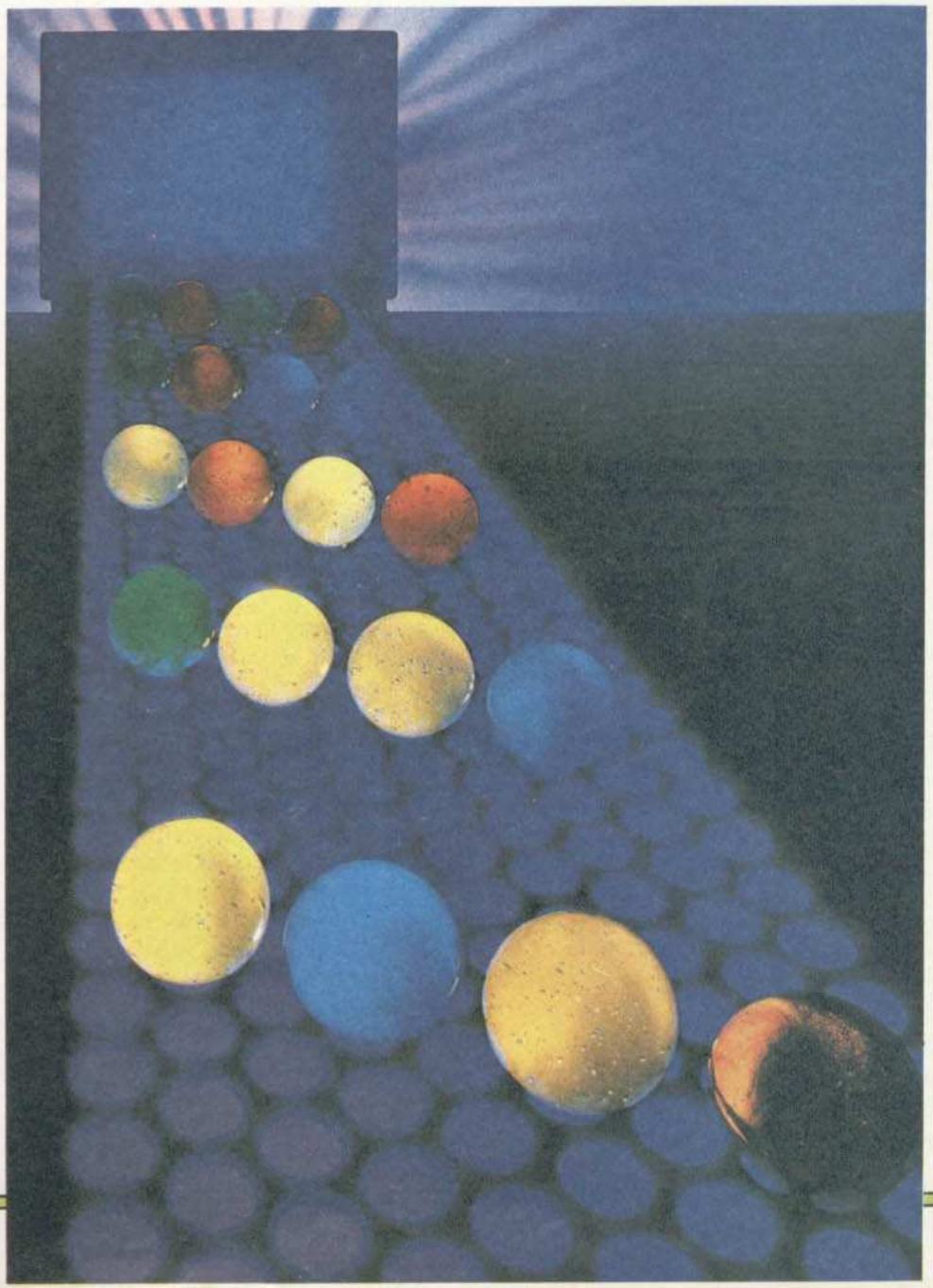
- AS REGRAS DO JOGO
- DEFINIÇÃO DAS CORES
- O COMPUTADOR DÁ AS DICAS
- ESCOLHA A MELHOR ESTRATÉGIA

Aceite o desafio lançado por seu microcomputador neste clássico jogo de lógica. Será que você é capaz de descobrir as cores sorteadas pela máquina, na ordem correta?

O objetivo de *Senha* é descobrir as quatro cores sorteadas pelo computador, entre as seis possíveis, além da ordem correta em que elas estão dispostas. Lembre-se de que pode ocorrer repetição de cores.

A cada tentativa, você deverá digitar as iniciais das cores — por exemplo, **RLCC** para roxo, laranja, cinza e cinza. O computador, então, responderá com um código, para ajudá-lo a decifrar o segredo. Para cada cor correta em posição errada ele imprimirá um caractere branco (ou a letra **B**), e quando a ordem estiver correta, o caractere terá a cor preta (ou a letra **P**). É lógico que a ordem dos códigos não corresponde à das cores; isto tornaria o jogo muito fácil. Você tem doze chances para decifrar a senha.

As cores usadas por cada programa são as seguintes: **A**marelo, a**Z**ul, azul-**C**laro, **V**ermelho, **M**agenta e **B**ranco para o micro Spectrum; **A**marelo, a**Z**ul, **V**ermelho, azul-**C**laro, **M**agenta e **L**aranja para o TRS-Color; **A**marelo, **V**ermelho, **R**oxo, **L**aranja, **M**agenta e **C**inza para o TK-2000, o Apple e o MSX.



```
10 BORDER 0: INK 0: PAPER 4:
CLS : LET N$="671254": DIM C(
4): DIM G(4): DIM F(4,2): LET
 C$="AZVCM"
14 PRINT AT 16,0;"CORES :"''"
A=AMARELO Z=AZUL C=AZUL CLARO
   V=VERMELHO  M=MAGENTA  B=B
RANCO"
15 FOR N=USR "A" TO USR "A"+7
: READ A: POKE N,A: NEXT N
17 DATA 0,24,60,126,126,60,24
,0
20 FOR K=1 TO 4: LET C(K)=VAL
 N$(INT (RND*5)+1): NEXT K:
LET G=1
30 INPUT "FACA A OPCAO ";B$
35 IF LEN B$<>4 THEN  GOTO 30
90 PRINT AT G,0;"OPCAO No. ";
G;AT G,14;: FOR K=1 TO 4: LET
G(K)=(7*(B$(K)="B"))+(6*(B$(K
)="A"))+(B$(K)="Z")+(2*(B$(K)
="V"))+(5*(B$(K)="C"))+(3*(B$
(K)="M"))
92 IF G(K)=0 THEN  LET K=4:
NEXT K: GOTO 30
95 PRINT  INK G(K); BRIGHT 1;
CHR$ 144;: IF K<>4 THEN
PRINT  BRIGHT 1;" ";
97 NEXT K
100 PRINT AT G,24;
110 LET N=0: LET R$=" ": FOR K
=1 TO 4: LET F(K,1)=0: LET F(K
,2)=0: IF G(K)=C(K) THEN  LET
R$=R$+" "+CHR$ 16+CHR$ 0+CHR$
144: LET F(K,1)=1: LET F(K,2)=
1: LET N=N+1
120 NEXT K
130 FOR K=1 TO 4: IF F(K,1)=1
THEN  GOTO 170
140 FOR J=1 TO 4: IF F(J,2)=1
THEN  GOTO 160
150 IF C(J)=G(K) THEN  LET R$=
```

```
R$+" "+CHR$ 16+CHR$ 7+CHR$ 144
: LET F(J,2)=1: LET J=4
160 NEXT J
170 NEXT K
180 PRINT AT G,23;R$: INK 0:
IF N=4 THEN  PRINT AT 21,0;"VO
CE ACERTOU APOS ";G;" TENTATIV
AS": GOTO 230
190 LET G=G+1: IF G<13 THEN
GOTO 30
200 PRINT "O CODIGO CORRETO ER
A ";: FOR K=1 TO 4: PRINT  INK
C(K);CHR$ 144;" ";: NEXT K
220 PRINT
230 PRINT "JOGA NOVAMENTE ?"
240 LET A$=INKEY$: IF A$=""
THEN  GOTO 240
250 IF A$="S" THEN  RUN : STOP
```

```
10 DIM C(3),G(3),F(3,1):C$="AZV
CML"
20 CLS:FOR K=0 TO 3:C(K)=RND(6)
:NEXT:G=1:PRINT @8,"adivinhe o
codigo"
30 PRINT @416,"TENTATIVA NO.";G
;"?  (EX: VZAC)":PRINT @448:B$=
""
40 A$=INKEY$:IF A$="" THEN 40
50 IF A$=CHR$(13) AND LEN(B$)=4
 THEN 90
60 IF A$=CHR$(8) AND LEN(B$)>0
THEN B$=LEFT$(B$,LEN(B$)-1)
70 IF LEN(B$)<4 AND INSTR(C$,A$
)<>0 THEN B$=B$+A$
80 PRINT @448,B$:GOTO 40
90 PRINT @32*G,"TENTATIVA";G;:F
OR K=0 TO 3:G(K)=INSTR(C$,MID$(
B$,K+1,1)):C=G(K)-(G(K)>3)
100 PRINT @32*G+11+K*2,CHR$(143
+C*16);:NEXT
110 N=0:R$="  ":FOR K=0 TO 3:F(
K,0)=0:F(K,1)=0:IF G(K)=C(K) TH
EN R$=R$+" "+CHR$(128):F(K,0)=1
:F(K,1)=1:N=N+1
120 NEXT
130 FOR K=0 TO 3:IF F(K,0)=1 TH
EN 170
140 FOR J=0 TO 3:IF F(J,1)=1 TH
EN 160
150 IF C(J)=G(K) THEN R$=R$+" "
+CHR$(207):F(J,1)=1:J=3
160 NEXT
170 NEXT
180 PRINT R$:IF N=4 THEN 200
190 G=G+1:IF G=13 THEN 210 ELSE
 30
200 PRINT @416," VOCE ACERTOU A
POS";G;"TENTATIVAS":GOTO 230
210 PRINT @416,"EU GANHEI. O CO
DIGO CORRETO ERA"
220 FOR K=0 TO 3:PRINT @448+K*2
," ";CHR$(143+C(K)*16-16*(C(K)>
3));:NEXT
230 PRINT:PRINT" JOGA NOVAMENTE
 ? (S/N)"
240 A$=INKEY$:IF A$<>"S" AND A$
<>"N" THEN 240
250 IF A$="N" THEN CLS:END ELSE
 20
```

```
10  DIM C(3),G(3),F(3,1):C$ = "
AVRLMC"
20  HOME : FOR K = 0 TO 3:C(K)
=  INT (6 *  RND (1) + 1): NEXT
 K
25 G = 1: PRINT  TAB( 6)"ADIVIN
HE A SENHA (CORES:AVRLMC)"
30  VTAB (23): PRINT "TENTATIVA
 NUMERO ";G;" (EX:AVRL):":B$ =
""
40  GET A$
50  IF A$ =  CHR$ (13) AND  LEN
 (B$) = 4 THEN 90
60  IF A$ =  CHR$ (8) AND  LEN
(B$) > 0 THEN B$ =  LEFT$ (B$,
LEN (B$) - 1)
70  IF  LEN (B$) < 4 THEN B$ =
B$ + A$
80  VTAB (G + 2): HTAB (21): PR
INT B$: GOTO 40
90  VTAB (G + 2): PRINT "CHANCE
 NUMERO ";G;
92  FOR K = 0 TO 3: FOR I = 1 T
O 6
94  IF  MID$ (B$,K + 1,1) =  MI
D$ (C$,I,1) THEN G(K) = I: GOTO
 100
96  NEXT I
98 G(K) = 0
100 C = G(K) - (G(K) > 3): NEXT
 K
105 N = 0:R$ = "  "
110  FOR K = 0 TO 3:F(K,0) = 0:
F(K,1) = 0: IF G(K) = C(K) THEN
 R$ = R$ + " " + "P":F(K,0) = 1
:F(K,1) = 1:N = N + 1
120  NEXT
130  FOR K = 0 TO 3: IF F(K,0)
= 1 THEN 170
140  FOR J = 0 TO 3: IF F(J,1)
= 1 THEN 160
150  IF C(J) = G(K) THEN R$ = R
$ + " " + "B":F(J,1) = 1:J = 3
160  NEXT J
170  NEXT K
180  HTAB (28): PRINT R$: IF N
= 4 THEN 200
190 G = G + 1: IF G = 13 THEN 2
10
195  GOTO 30
200  VTAB (22): PRINT " VOCE AC
ERTOU A SENHA EM ";G;" CHANCES"
: GOTO 230
210  VTAB (22): PRINT " EU GANH
EI, A SENHA ERA ";
220  FOR K = 0 TO 3: PRINT  MID
$ (C$,C(K),1);: NEXT
230  HTAB (1): VTAB (23): PRINT
 " QUER JOGAR NOVAMENTE?(S/N)
 "
240  GET A$
250  IF A$ = "S" THEN  GOTO 20
260  END
```

```
10 DIM C(3),G(3),F(3,1):C$="VAR
LMC"
20 CLS:KEY OFF:FOR K=0 TO 3:C(K
)=INT(6*RND(-TIME)+1):NEXT:G=1:
```

Como chegar mais rapidamente à seqüência correta das cores?

Existe apenas um segredo: obter o máximo de informações em cada jogada. Com seis tentativas, por exemplo, utilizando somente uma cor em cada, você descobrirá facilmente as quatro cores da senha, mas nada saberá sobre suas posições. Assim, convém ter sempre em vista ambos os objetivos. Talvez você perca mais tempo para chegar às cores corretas, mas a definição de suas posições na seqüência será bem mais rápida.

```
PRINT TAB(4) "ADIVINHE A SENHA
(CORES:VARLMC)"
30 LOCATE 3,22: PRINT "TENTATIV
A NO";G; "? (EX:AVRL)":B$=""
40 A$=INKEY$:IF A$="" THEN 40
50 IF A$=CHR$(13) AND LEN(B$)=4
 THEN 90
60 IF A$=CHR$(8) AND LEN(B$)>0
THEN B$=LEFT$(B$,LEN(B$)-1)
70 IF LEN(B$)<4 AND INSTR(C$,A$
)<>0 THEN B$=B$+A$
80 LOCATE 24,G+4:PRINT B$:GOTO
40
90 LOCATE 3,G+4:PRINT"Tentativa
 número ";G;:FOR K=0 TO 3:G(K)=
INSTR(C$,MID$(B$,K+1,1)):C=G(K)
-(G(K)>3):NEXT K
110 N=0:R$="  ":FOR K=0 TO 3:F(
K,0)=0:F(K,1)=0:IF G(K)=C(K) TH
EN R$=R$+" "+"P":F(K,0)=1:F(K,1
)=1:N=N+1
120 NEXT
130 FOR K=0 TO 3:IF F(K,0)=1 TH
EN 170
140 FOR J=0 TO 3:IF F(J,1)=1 TH
EN 160
150 IF C(J)=G(K) THEN R$=R$+" "
+"B":F(J,1)=1:J=3
160 NEXT
170 NEXT
180 LOCATE 28,G+4:PRINT R$:IF N
=4 THEN 200
190 G=G+1:IF G=13 THEN 210 ELSE
 30
200 LOCATE 1,21:PRINT"Você acer
tou a senha em ";G;" chances":G
OTO 230
210 LOCATE 1,21:PRINT "Eu venci
  a senha correta era ";:
220 FOR I=0 TO 3:PRINT MID$(C$,
C(I),1);:NEXT
230 LOCATE 1,22:PRINT"Quer joga
r novamente? (S/N)    "
240 A$=INKEY$:IF A$<>"S" AND A$
<>"N" THEN 240
250 IF A$="N" THEN CLS:END:ELSE
 20
```

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.

NO PRÓXIMO NÚMERO

APLICAÇÕES

Complete sua planilha eletrônica. Como entrar as equações. Cópias absolutas e relativas. Instruções especiais.

CÓDIGO DE MÁQUINA

Avalanche: início da caminhada. Onde Willie está pisando? Imobilidade. Passos fatais. O último suspiro.

PROGRAMAÇÃO BASIC

Páginas gráficas e exigência de espaço. Técnicas de paginação. Armazenagem e recuperação. Animações gráficas.