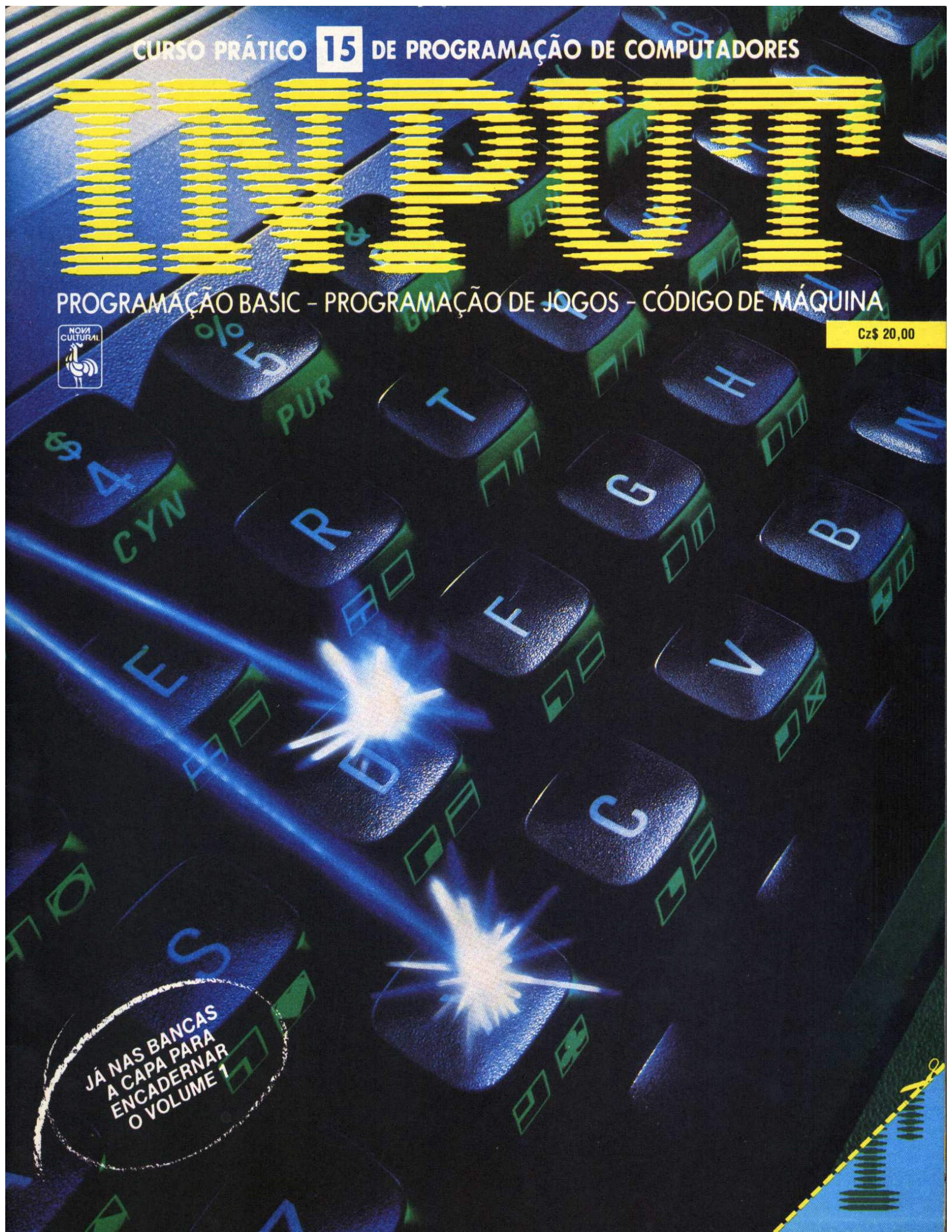
CURSO PRÁTICO 15 DE PROGRAMAÇÃO DE COMPUTADORES

BASIC

PROGRAMAÇÃO BASIC - PROGRAMAÇÃO DE JOGOS - CÓDIGO DE MÁQUINA

NOVA CULTURAL

Cz$ 20,00



JÁ NAS BANCAS
A CAPA PARA
ENCADERNAR
O VOLUME 1

INPUT

Vol. 1 N.º 15

NESTE NÚMERO

APLICAÇÕES

MELHORE A SUA DATILOGRAFIA

Incorpore números e sinais de pontuação ao seu aprendizado de datilografia e aprenda a utilizar a tecla <**SHIFT**> 281

PERIFÉRICOS

JOYSTICKS

Versátil e barato, o joystick pode ser usado tanto para animar jogos como em aplicações mais "sérias" 287

PROGRAMAÇÃO BASIC

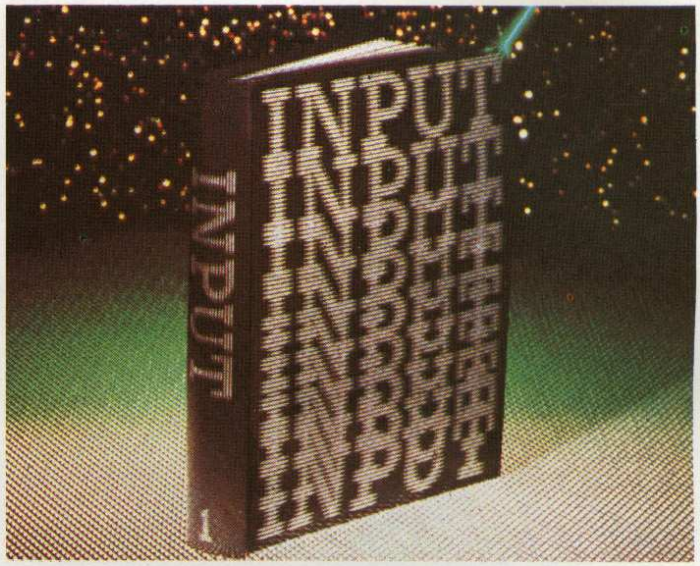
ORDENAÇÃO PELO MÉTODO DE BOLHAS

Como globos de ar em ascensão no interior de um líquido. Como escrever uma sub-rotina de ordenação. Defina as variáveis 292

CÓDIGO DE MÁQUINA

ASSEMBLER PARA O TRS-COLOR

Conversão automática da linguagem Assembly para código de máquina 296



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); Av. Industrial, 117 (Santo André); e, no Rio de Janeiro: Rua da Passagem, 93 (Botafogo). **2. Por carta** — Poderão ser solicitados exemplares atrasados também por carta, que deve ser enviada para DINAP — Distribuidor Nacional de Publicações — Números Atrasados — Estrada Velha de Osasco, 132 (Jardim Tereza) — CEP 06000 — Osasco — São Paulo. **3. Por telex** — Utilize o nº (011) 33670 ABSA. Em **Portugal**, os pedidos devem ser feitos à Distribuidora Jardim de Publicações Ltd. — Qta. Pau Varais, Azinhaga de Fetais — 2685, Camarate — Lisboa; Tel. 257-2542 — Apartado 57 — Telex 43 069 JARLIS P.
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. **Atenção**: Após seis meses do encerramento da coleção, os pedidos serão atendidos, dependendo da disponibilidade de estoque. **Obs.:** Quando pedir livros, mencione sempre o título e/ou o 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

Diretora Editorial: Iara Rodrigues

Editor chefe: Paulo de Almeida
Editor de texto: Cláudio A.V. Cavalcanti
Editor de Arte: Eduardo Barreto
Chefe de Arte: Carlos Luiz Batista
Assistentes de Arte: Ailton Oliveira Lopes, Dilvacy M. Santos, José Maria de Oliveira, Grace A. 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
Secretário Gráfico: Antonio José Filho

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: Maria Fernanda Sabbatini
Adaptação, programação e redação: Abílio Pedro Neto, Aluísio J. Dornellas de Barros, Marcelo R. Pires Therezo, Raul Neder Porrelli
Coordenação geral: Rejane Felizatti Sabbatini
Editora de Texto: Ana Lúcia B. de Lucena
Assistente de Arte: Dagmar Bastos Sampaio
COMERCIAL
Diretor Comercial: Roberto Martins Silveira
Gerente Comercial: Flávio Ferrucio Maculan
Gerente de Circulação: Denise Maria Mozol

PRODUÇÃO
Gerente de Produção: João Stungis
Coordenador de Impressão: Atilio Roberto Bonon
Preparador de Texto/Coordenador: Eliel Silveira Cunha

Preparadores de Texto: Ana Maria Dilguerian, Antonio Francelino de Oliveira, Karina Ap. V. Grechi, Levon Yacubian, Maria Teresa Galluzzi, Paulo Felipe Mendrone
Revisor/Coordenador: José Maria de Assis
Revisoras: Conceição Aparecida Gabriel, Isabel Leite de Camargo, Ligia Aparecida Ricetto, Maria do Carmo Leme Monteiro, Maria Luiza Simões, Maria Teresa Martins Lopes.

Edição organizada pela Editora Nova Cultural Ltda.
(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.

MELHORE A SUA DATILOGRAFIA

- ACRESCENTE NÚMEROS
- A TECLA <SHIFT>
- DESENVOLVA UM RITMO REGULAR
- VELOCIDADE E PRECISÃO

Divirta-se com um joguinho de ação rápida e aprenda ao mesmo tempo a trabalhar com as teclas de números, ampliando a sua habilidade como datilógrafo.

Depois de ter estudado as duas primeiras lições de datilografia, você já está provavelmente familiarizado com as teclas de letras e com os sinais de pontuação das três fileiras inferiores do teclado. Até agora, porém, não tocamos nas importantes teclas numéricas (muito usadas, aliás, na digitação de programas.)

Da mesma forma, ainda não ensinamos as formas de manipulação das teclas para se obter letras maiúsculas ou minúsculas, nem explicamos os sinais de pontuação adicionais e demais símbolos disponíveis no teclado.

Esta lição será dedicada às teclas ainda não estudadas. Nela, apresentaremos também um pequeno e divertido exercício, que fará com que você melhore a sua velocidade, precisão e ritmo no trabalho com o teclado.

ACRESCENTANDO OS NÚMEROS

Para treinarmos a datilografia com a fileira do teclado que contém os algarismos, basta fazer algumas modificações no programa apresentado nas lições anteriores do curso (páginas 253 e 276).

Desta forma, carregue o último programa utilizado e digite as linhas adicionais, expostas a seguir. Como você verá, algumas delas irão apenas substituir linhas já existentes, ao passo que outras serão adicionadas ao programa:

```
 30 LET S$="1A2S3D4F5G6H7J8K9L
 0"
210 FOR K=6 TO 24
230 LET R$=S$(K-5)
320 LET RN=INT (RND*19) 1
330 PRINT AT 10,RN+5;"*": LET
R$=S$(RN)
350 PRINT AT 10,RN+5;" "
 440 LET RN=INT (RND*19)+1
 530 PRINT AT 10,13;"
 ": PRINT AT 10,13;T$
 540 FOR M=1 TO LEN T$: PRINT
 AT 9,11+M;" *       "
 610 FOR N=1 TO 5: RESTORE :
 LET RN=INT (RND*24)+1: FOR K=1
 TO RN: READ X$: NEXT K
1010 PRINT AT 12,6;S$
2000 DATA "MC6809E","VALOR","UL
TIMO","Z80A","RELAXE","6502","A
37XZ","1024","VASTO","JUNCO","R
ETORNO","67VDG"
2010 DATA "APENAS","AGITADO","7
4LS83","REDONDO","LINEAR","1986
","30123","CONGELADO","4MHZ","W
X101","64MB","VIDEO"
```

```
10 OB$="1A2S3D4F5G6H7J8K9L0:-;"
210 AP=1252
220 FOR K=1 TO 22
320 AP=1252+RND(22)
430 P$=MID$(OB$,RND(22),1)
1020 PRINT @261,OB$
9000 DATA MC6809E,VALOR,"ULTIMO
:",Z80A,RELAXE,6502,A37XZ,-1024
,VASTO,JUNCO
9010 DATA RETURN,A847VDG,145,22
,"APENAS,",AGITADO,74LS83,-93.4
1,REDONDO,LINEAR
9020 DATA PROCESSO,CONGELADO,TR
AZ,1986,DESENHO,SANGUE,842.52,"
301,123",350KG
```

```
10 OB$ = "1A2S3D4F5G6H7J8K9L0;:
-"
210 AP = 8
220  FOR K = 1 TO 22:AP = AP +
1: GOSUB 1100: NEXT
320 AP = 9 +  INT ( RND (1) * 2
3)
420  FOR K = 1 TO 20:P$ =  MID$
 (OB$, INT ( RND (1) * 22) + 1,
1)
640 P = 1: VTAB 11: HTAB 2: PRI
NT P$
660  VTAB 15: HTAB P + 1: PRINT
  MID$ (P$,P,1)
1010  VTAB 12: HTAB 9: PRINT OB
$
9000  DATA  MC6809E,VALOR,"ULTI
MO:",Z80A,RELAXE,6502,A37XZ,-10
24,VASTO,JUNCO
9010  DATA  RETURN,A847VDG,145.
22,"APENAS,",AGITADO,74LS83,-93
.41,REDONDO,LINEAR
9020  DATA   PROCESSO,CONGELADO
,TRAZ,1986,DESENHO,SANGUE,842.5
2,"301,23",350KG
```

```
10 OB$="1A2S3D4F5G6H7J8K9L0Ç-="
:Z$=CHR$(219):S$="L10 O2 G"
210 AP=355
220 FOR K=1 TO 22
320 AP=356+INT(RND(1)*22)
430 P$=MID$(OB$,INT(RND(1)*22)+
1,1)
1010 LOCATE 2,12:PRINTOB$
9000 DATA MC6809E,VALOR,"ULTIMO
:",Z80A,RELAXE,6502,A37XZ,-1024
,VASTO,JUNCO
9010 DATA RETURN,A847VDG,145.22
,"SOMENTE,",AGITAR,74LS83,-93.4
1,CIRCULO,LINEAR
9020 DATA PROCESSO,FRIO,TRAZ,19
86,DESENHO,SANGUE,842.52,CAÇA,3
50KG
```

Quando o programa for rodado, um menu contendo os cinco níveis já conhecidos será apresentado na tela; você deverá então optar por um deles. Como das vezes anteriores, existirão muitas combinações possíveis, dependendo do

número de teclas a serem incluídas nos exercícios. Nos níveis mais baixos de dificuldade, o computador proporá exercícios em que as teclas numéricas e, algumas vezes, de pontuação serão combinadas aleatoriamente com as outras teclas, que você já conhece das duas primeiras lições. Essa combinação dificultará as coisas para você, e o obrigará a não se concentrar apenas nas teclas numéricas.

Nos níveis mais altos de dificuldade, o computador pedirá que você digite uma mistura de palavras, grupos de números e combinações de letras e algarismos. As palavras e números selecionados se encontram nas declarações **DATA**, quase no final do programa. Eles podem ser trocados, caso você queira aumentar o desafio depois de se familiarizar bem com os exercícios originais. Entretanto, lembre-se de manter o mesmo número de palavras (ou grupo de caracteres); do contrário, ocorrerá um erro de execução quando o programa não encontrar o número correto de itens em **DATA**.

Uma vez dominadas as teclas numéricas, passe para a lição seguinte, cujo objetivo é praticar com os caracteres acessíveis apenas por intermédio da tecla <**SHIFT**>.

AS LETRAS MAIÚSCULAS

A tecla <**SHIFT**> corresponde à alavanca de maiúsculas de uma máquina de escrever comum. Em alguns computadores, essa função também pode ser desempenhada pela <**CAPS LOCK**>. Para aprender a usá-las, adicione as linhas abaixo à última versão do programa (novamente, algumas linhas serão inteiramente substituídas):

```
 20 POKE 23658,0: LET ER=0
 30 LET S$="A!aS@sD#dF$fG%gH&h
 J'jK(kL)I0"
210 FOR K=2 TO 29
230 LET R$=S$(K-1)
320 LET RN=INT (RND*28)+1
330 PRINT AT 10,RN+1;"*": LET
R$=S$(RN)
350 PRINT AT 10,RN+1;" "
440 LET RN=INT (RND*28)+1
610 FOR N=1 TO 4: RESTORE :
LET RN=INT (RND*24)+1: FOR K=1
TO RN: READ X$: NEXT K
1010 PRINT AT 12,2;S$
2000 DATA "$235.50","PRINT#","&
H1200","23.5%","Conta","LONDRES
","Eles","15@&12","(abaixo)","H
+9=1D","**Obs**","Fogo!!"
2010 DATA ";-;;-;","Extra","Bei
ja-flor","Lugar","4*4=16","Quem
?","6:10pm","Nos","$15.40","Dir
igir","ATENCAO","100/4"
```

```
10 OB$="!A"+CHR$(34)+"S#D$F%G&H
'J(K)L*=+"
20 POKE 282,0:CLS
210 AP=1250
220 FOR K=1 TO 21
320 AP=1250+RND(21)
430 P$=MID$(OB$,RND(21),1)
999 POKE 282,255:CLS:END
1020 PRINT @259,OB$
9000 DATA PRINT#,Mostre,&H4000,
Fora!!,(abaixo),H+9=1D,$500.10,
D/100%,Eles,**obs**
9010 DATA Extra,Carga,DIARIO,Co
nta,Mes,Resposta,Hoje,Gerente,S
etor
9020 DATA LONDRES,Concha,Toque,
;-;;-;,Sucesso,Beija-flor,Lugar
,Campo,Direto
```

```
10 OB$ = "!A" +  CHR$ (34) + "S
#D$F%G&H'J(K)L+*=<>?"
220  FOR K = 1 TO 24:AP = AP +
1: GOSUB 1100: NEXT
320 AP = 9 +  INT ( RND (1) * 2
5)
```

```
420  FOR K = 1 TO 20:P$ =  MID$
 (OB$, INT ( RND (1) * 24) + 1,
1)
9000  DATA  PR#,Mostre,$H4000,F
ora!!,(abaixo),H + 9 = 1D,Cz$ 5
00.10,D/100%,Eles,**Obs**
9010  DATA  Extra,Carga,Diario,
Conta,Mes,Hoje,Resposta,Gerente
,Setor
9020  DATA  LONDRES,Concha,Toqu
e,;-;;-;,Sucesso,Beija-flor,Lug
ar,Campo,Direto
```

```
10 OB$="!A@S#D$F%G"+CHR$(34)+"H
&J*K(L)Ç_+?>":Z$=CHR$(219):S$="
L10 O2 G"
220 FOR K=1 TO 24
320 AP=356+INT(RND(1)*24)
430 P$=MID$(OB$,INT(RND(1)*24)+
1,1)
9000 DATA PRINT#,Mostra,&H4000,
Fora!!,(abaixo),H + 9 = 1D,Cz$
500.10,D/100%,Eles,** Obs **
9010 DATA Extra,Cargo,DIÁRIO,Co
nta,Mês,Resposta,Hoje,Gerente,S
eção
9020 DATA LONDRES,Concha,Toque,
;?;?,Sucesso,Beija-flor,Lugar,C
ampo,Guia
```

O TRS-Color não imprime letras minúsculas na tela. Em vez disso, exibe-as em vídeo inverso (letras claras sobre fundo escuro).

Nos micros da linha Apple que dispõem de minúsculas, mude a tecla seletora para minúsculas ao digitar as linhas **DATA**. Infelizmente, os micros da linha Apple II+ padrão, que não contam com letras minúsculas, não conseguirão rodar o programa a seguir. Nos micros da linha MSX, certifique-se de que a tecla <**CAPS**> está desativada.

Os níveis mais baixos de dificuldade do programa apresentam agora os caracteres que só estão disponíveis quando as teclas são pressionadas simultaneamente com o comando <**SHIFT**>: pontuação, símbolos matemáticos, etc... Eles estão misturados com as letras da fileira central do teclado, o que significa que você deve retornar a eles toda vez que pressioná-los.

Nos níveis mais altos são apresentados palavras e grupos de caracteres de uma lista interna, tal como antes; agora, porém, você encontrará as letras maiúsculas e outros símbolos que só podem ser obtidos com o <**SHIFT**>, misturados com as letras minúsculas.

Conhecidos esses exemplos particulares de teste, substitua as declarações **DATA** por um novo conjunto de dados (lembre-se de manter o mesmo número total de palavras).

Passe para a próxima seção apenas quando estiver encontrando — sem hesitação e sem olhar para o teclado — todos os caracteres com que treinou.

JOGO DE VELOCIDADE

Um dos modos mais eficazes de se melhorar a precisão e a velocidade de datilografia consiste em pressionar as teclas ao compasso de um metrônomo (instrumento que serve para regular os andamentos musicais). Este deve ser acelerado à medida que o movimento de seus dedos se tornar mais rápido e preciso.

Mas por que dar-se ao trabalho de utilizar um metrônomo, quando o seu computador possui um relógio embutido? O próximo programa é um novo e completo exercício de digitação, planejado como um jogo, onde a contagem depende do seu desempenho no teclado. Ele é composto por duas partes. A primeira, exibe uma linha de caracteres selecionada aleatoriamente — você tem que digitar, em seqüência, os caracteres à medida que forem aparecendo. A segunda parte é mais difícil — agora, os caracteres são lançados aleatoriamente na tela, um por um; deste modo, você não tem idéia do que virá a seguir.

Antes de iniciar o teste, selecione o seu próprio nível de dificuldade. Isso deve ser feito dizendo-se ao computador com que velocidade você quer que as letras sejam exibidas — em outras palavras, quantos caracteres por minuto você quer digitar. Então, o computador estabelecerá um tempo limitado, dentro do qual você deve digitar cada caractere; do contrário, será emitida uma contagem de erros. No primeiro nível, isso é feito por intermédio de um indicador móvel que mostra qual letra deveria estar sendo digitada; no segundo nível, o caractere é iluminado por algum tempo.

Ao começar o primeiro teste, você pode escolher se quer o teclado normal (com letras, apenas) ou teclado completo (com todos os símbolos). E antes de passar para o segundo nível (o teste dos caracteres) você deve decidir quanto tempo é capaz de sustentá-lo. O computador perguntará quantos caracteres, no

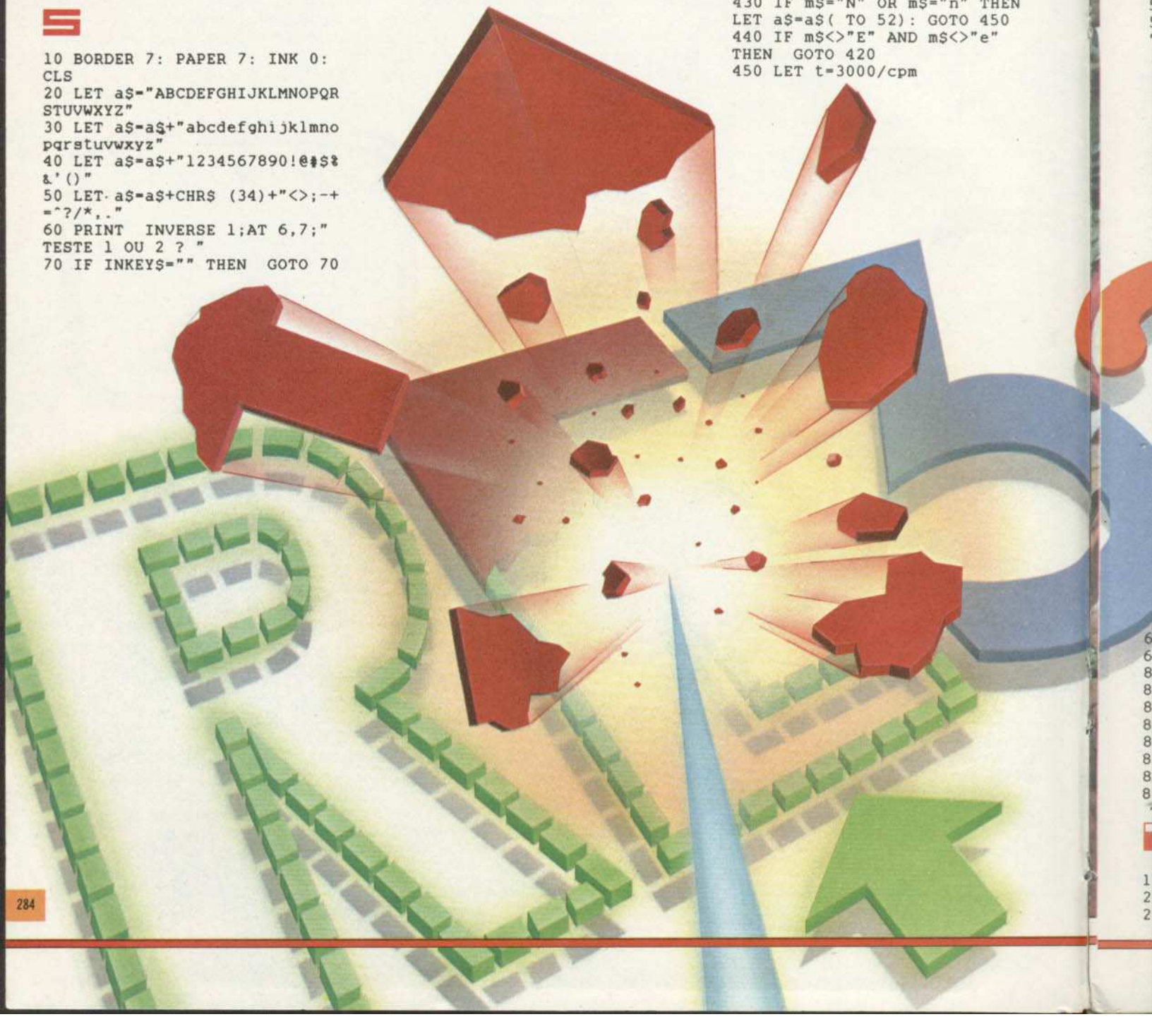
total, você quer que façam parte do teste.

Imediatamente depois de ter exibido todos os caracteres, o computador parará e fornecerá uma contagem baseada nos seus erros.

Não é apenas a velocidade total de digitação no computador que conta nesse teste de desafio: você precisará também desenvolver um ritmo constante. Para ajudá-lo a adquirir esse ritmo, o computador dará um sinal sonoro assim como um sinal visual de prontidão para cada letra.

Agora digite o programa propriamente dito, e teste a sua habilidade como datilógrafo:

```
10 BORDER 7: PAPER 7: INK 0:
CLS
20 LET a$="ABCDEFGHIJKLMNOPQR
STUVWXYZ"
30 LET a$=a$+"abcdefghijklmno
pqrstuvwxyz"
40 LET a$=a$+"1234567890!@#$%
&'()"
50 LET a$=a$+CHR$ (34)+"<>;-+
=^?/*,."
60 PRINT  INVERSE 1;AT 6,7;"
TESTE 1 OU 2 ? "
70 IF INKEY$="" THEN  GOTO 70
 80 LET i$=INKEY$: IF i$="2"
 THEN  GOTO 400
 90 IF i$<>"1" THEN  GOTO 70
100 CLS : INPUT "Quantos carac
teres por minuto? ";cpm
110 LET t=3000/cpm
120 LET s$=""
130 FOR n=1 TO 30
140 LET s$=s$+a$(INT (RND*84)+
1)
150 NEXT n
160 PRINT  BRIGHT 1;AT 11,1;s$
200 GOSUB 800: LET er=0: FOR r
=1 TO 30
210 POKE 23672,0: POKE 23673,0
220 PRINT AT 10,r-1;" *"
230 SOUND .02,20
240 IF  PEEK  23672+256*PEEK
23673>=t THEN  LET er=er+1:
GOTO 300
250 LET i$=INKEY$: IF i$=""
THEN  GOTO 240
260 IF i$=s$(r) THEN  PRINT AT
12,r;"^": GOTO 280
270 LET er=er+1
280 IF  PEEK  23672+256*PEEK
23673<t THEN  GOTO 280
300 NEXT r
310 PRINT AT 16,3;"VOCE ERROU
";er;" DAS 30"
320 FOR f=1 TO 200: NEXT f
330 GOTO 20
400 CLS : INPUT "Numero de tec
las por minuto? ";cpm
410 INPUT "Numero de caractere
s?";r
420 INPUT "(N)ormal ou (E)sten
dido? "; LINE m$
430 IF m$="N" OR m$="n" THEN
LET a$=a$( TO 52): GOTO 450
440 IF m$<>"E" AND m$<>"e"
THEN  GOTO 420
450 LET t=3000/cpm
```

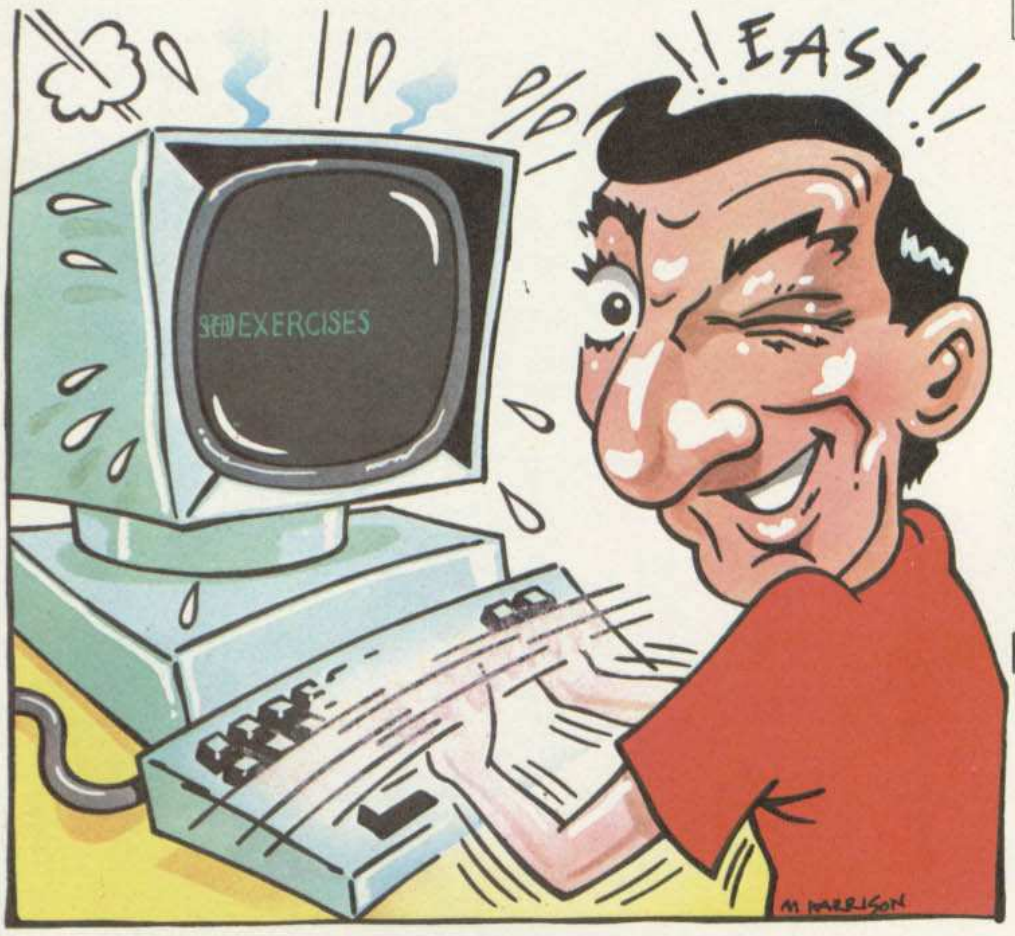


```
460 GOSUB 800
470 LET er=0
480 FOR n=1 TO r
490 POKE 23672,0: POKE 23673,0
500 LET r$=a$(INT (RND*LEN a$)
+1)
510 PRINT  INVERSE 1;AT 10,15;
r$; INVERSE 0;AT 11,15;" "
520 SOUND .02,20
530 IF  PEEK  23672+256*PEEK
23673>=t THEN  LET er=er+1:
GOTO 580
540 LET i$=INKEY$: IF i$=""
THEN  GOTO 530
550 IF i$=r$ THEN  PRINT AT 11
,15;"^": GOTO 570
560 LET er=er+1
570 IF PEEK 23672+256*PEEK
23673<t THEN  GOTO 570
580 NEXT n
590 PRINT AT 16,3;"VOCE ERROU
";er;" DAS ";n-1
600 FOR f=1 TO 200: NEXT f
610 GOTO 20
800 LET c$="5..4..3..2..1..0"
810 FOR n=1 TO 16
820 PRINT AT 2,5+n;c$(n)
830 PAUSE 10
840 NEXT n
850 SOUND .2,10
860 PRINT AT 2,0;TAB 31;" "
870 RETURN
```

```
10 CLS
20 PRINT @70,"QUAL TESTE (1 OU
2)?"
30 PRINT @102,"DIGITE (0) PARA
SAIR"
40 A$=INKEY$:IF A$<"0" OR A$>"2
" THEN 40
50 ON VAL (A$)+1 GOSUB 1000,600
,200
60 POKE 282,255
70 ER=0:W$="":B$=""
80 GOTO 20
200 CLS:INPUT"DIGITE QUANTAS TE
CLAS PRESSIONADAS POR MINUTO";K
P
210 IF KP<1 THEN 200
220 INPUT "DIGITE O NUMERO DE C
ARACTERES ";NC
230 IF NC<1 THEN 220
240 NM=NC
250 PRINT:PRINT"NORMAL OU ESTEN
DIDO (N/E)?"
260 A$=INKEY$:IF A$<>"N" AND A$
<>"E" THEN 260
270 RN=90:ST=32:IF A$="N" THEN
RN=58:ST=64
280 POKE 282,0
290 TM=3000/KP
300 CLS0:PRINT" PRESSIONE A TEC
LA DEPOIS DO BIP "
310 PRINT @238,"    ";:PRINT @27
0,"    ";:PRINT @302,"    ";
320 W$=CHR$(RND(RN)+ST)
330 IF W$>"Z" AND W$<"a" THEN 3
20
340 PRINT @271,W$;
350 TIMER=0
360 A$=INKEY$:IF A$="" THEN 380
370 B$=A$:IF B$=W$ THEN SCREEN
0,1
380 IF TIMER <TM THEN 360
390 SOUND 150,1:IF B$="" THEN 3
50
400 W$=CHR$(RND(RN)+ST)
410 IF W$>"Z" AND W$<"a" THEN 4
00
420 PRINT @271,W$;
430 TIMER=0
440 A$=INKEY$:IF A$="" THEN 460
450 B$=A$:IF B$=W$ THEN SCREEN
0,1
460 IF TIMER<TM THEN 440
470 SOUND 150,1
480 IF B$<>W$ THEN ER=ER+1:GOTO
 490
490 POKE 1295,128
500 NC=NC-1:IF NC>0 THEN 400
510 CLS:PRINT @448,"COM";KP;"TE
CLAS PRESSIONADAS POR MINUTO"
520 PRINT "VOCE ERROU";ER;"DAS"
;NM
530 RETURN
600 CLS:INPUT"DIGITE QUANTAS TE
CLAS PRESSIONADAS POR MINUTO";K
P
610 IF KP<1 THEN 600
620 PRINT:PRINT"NORMAL OU ESTEN
DIDO (N/E)?"
630 A$=INKEY$:IF A$<>"N" AND A$
<>"E" THEN 630
640 RN=91:ST=31:IF A$="N" THEN
RN=58:ST=64
650 TM=3000/KP
660 CLS:POKE 282,0
670 FOR K=1 TO 32
680 CR=RND(RN)+ST
690 IF CR>90 AND CR<97 THEN CR=
32
700 W$=W$+CHR$(CR)
710 NEXT
720 AP=1248
730 POKE AP,106
740 PRINT @256,W$
750 TIMER=0
760 A$=INKEY$:IF A$="" THEN 780
770 B$=A$
780 IF TIMER<TM THEN 760
790 SOUND 150,1:IF B$="" THEN 7
50
800 TIMER=0
810 A$=INKEY$:IF A$="" THEN 830
820 B$=A$
830 IF TIMER<TM THEN 810
840 SOUND 150,1
850 IF B$<>MID$(W$,AP-1247,1) T
HEN ER=ER+1:GOTO 860
855 POKE AP+64,94
860 POKE AP,96
870 AP=AP+1
```

```
880 IF AP=1280 THEN 910
890 POKE AP,106
900 B$="":GOTO 800
910 CLS:PRINT @481,"COM";KP;"TE
CLAS POR MINUTO"
920 PRINT"VOCE ERROU";ER;"DAS 3
2";:RETURN
1000 CLS
```

```
10 CLS:Z$=CHR$(219):R=RND(-TIME
)
20 LOCATE 5,3:PRINT"QUAL TESTE?
 (1-2)"
30 LOCATE 5,4:PRINT"DIGITE <0>
PARA TERMINAR"
40 A$=INKEY$:IFA$<"0"ORA$>"2"TH
EN40
50 ON VAL(A$)+1 GOSUB 1000,600,
200
60 ER=0:W$="":B$=""
70 GOTO20
200 CLS:INPUT"TOQUES POR MINUTO
";KP
210 IFKP<1THEN200
220 INPUT"NÚMERO DE CARACTERES"
;NC
230 IFNC<1THEN220
240 NM=NC
250 PRINT:PRINT"TECLADO: LETRAS
 OU TOTAL (L/T)?"
260 A$=INKEY$:IFA$<>"L"ANDA$<>"
T"THEN260
270 RN=90:ST=33:IFA$="L"THENRN=
58:ST=65
280 TM=3600/KP
300 CLS:PRINT"PRESSIONE A TECLA
 APÓS O BIP"
310 LOCATE15,10:PRINTSTRING$(4,
219)
320 LOCATE15:PRINTZ$;"  ";Z$
330 LOCATE15:PRINTZ$;"  ";Z$
340 LOCATE15:PRINTSTRING$(4,219
)
350 W$=CHR$(RND(1)*RN+ST)
360 IF W$<"Z"ANDW$>"a"THEN350
370 LOCATE17,12:PRINTW$;:TIME=0
380 A$=INKEY$:IFA$=""THEN390ELS
EB$=A$:IFB$=W$THENCOLOR15,6
390 IFTIME<TMTHEN380ELSEBEEP:IF
B$=""THEN370
400 IFB$<>W$THENER=ER+1
410 COLOR 15,4,4
420 NC=NC-1:IFNC>0THEN350
430 CLS:LOCATE2,12:PRINT"A";KP;
"toques por minuto, você errou"
;ER;"em";NM
440 RETURN
600 CLS:INPUT"TOQUES POR MINUTO
";KP
610 IFKP<1THEN600
620 PRINT:PRINT"TECLADO: LETRAS
 OU TOTAL? (L/T)"
630 A$=INKEY$:IFA$<>"L"ANDA$<>"
T"THEN630
640 RN=91:ST=32:IFA$="L"THENRN=
58:ST=65
650 TM=3600/KP
660 CLS:FORK=1TO32
670 CR=INT(RND(1)*RN)+ST
```



MICRO DICAS

COMO AUMENTAR A VELOCIDADE DE DIGITAÇÃO

Convém começar a execução do programa de jogo de velocidade com o teclado normal e a uma velocidade baixa (por exemplo, de cerca de trinta a cinqüenta caracteres por minuto). Ao mesmo tempo, procure não alterar o ritmo de trabalho, digitando com a mesma velocidade tanto os caracteres conhecidos como os que lhe são menos familiares (esse conselho vale principalmente para quando você estiver com o teclado completo).

Se o seu ritmo se mantiver constante, você poderá selecionar o teclado completo e aumentar, gradativamente, a velocidade de digitação.

Outra coisa muito importante para quem está aprendendo a datilografar é fazer todos os exercícios propostos *sem olhar* para o teclado do computador.

Mantenha os seus olhos fixos todo o tempo na tela à sua frente, e repouse as mãos sobre o teclado, distribuindo os dedos pelas teclas de apoio da fileira central, conforme ensinamos na primeira lição.

```
680 IFCR>90ANDCR<97THENCR=32
690 W$=W$+CHR$(CR)
700 NEXT
710 AP=564
720 VPOKEBASE(0)+AP,205
730 LOCATE3,15:PRINTW$
740 TIME=0
750 A$=INKEY$:IFA$=""THEN760ELS
EB$=A$
760 IFTIME<TMTHEN750
770 BEEP:IFB$=""THEN740
780 IFB$<>MID$(W$,AP-563,1)THEN
ER=ER+1ELSEVPOKEBASE(0)+AP+80,1
790 VPOKEBASE(0)+AP,0:AP=AP+1:I
FAP=596THEN810
800 VPOKEBASE(0)+AP,205:GOTO740
810 CLS:LOCATE2,15:PRINT"A";KP;
"toques por minuto, você errou"
;ER;"de 32"
820 RETURN
1000 CLS
```

FAÇA TODOS OS EXERCÍCIOS

Para adquirir um conhecimento profundo de todas as teclas de caracteres pratique todos os exercícios apresentados até agora. Esta parte do curso é independente. Entretanto, em um artigo posterior, você terá oportunidade de praticar suas habilidades com algumas frases e sentenças.

JOYSTICKS

Sistema de controle dos mais versáteis e baratos, o joystick é um periférico que serve tanto para aplicações "sérias" como para tornar os jogos mais interessantes.

- COMUNIQUE-SE COM O COMPUTADOR
- COMO UTILIZAR UM JOYSTICK
- TIPOS DE JOYSTICKS
- ESCOLHA O JOYSTICK CERTO

Embora estejam ficando cada vez mais sofisticados, os computadores domésticos ainda não conseguem dar instruções a si mesmos. Por enquanto (e esperamos que para sempre...), a responsabilidade disso recai inteiramente sobre o programador ou o usuário. Aprender a programar é, portanto, aprender a se comunicar com o computador.

O principal meio de comunicação do usuário com o computador continua sendo o teclado. Mas existem muitas razões para considerar esse meio como longe do ideal: a principal delas é que o teclado torna a comunicação comparativamente lenta em relação a outros métodos. Além disso, o uso eficiente do teclado exige que o usuário aprenda a digitar com rapidez e precisão.

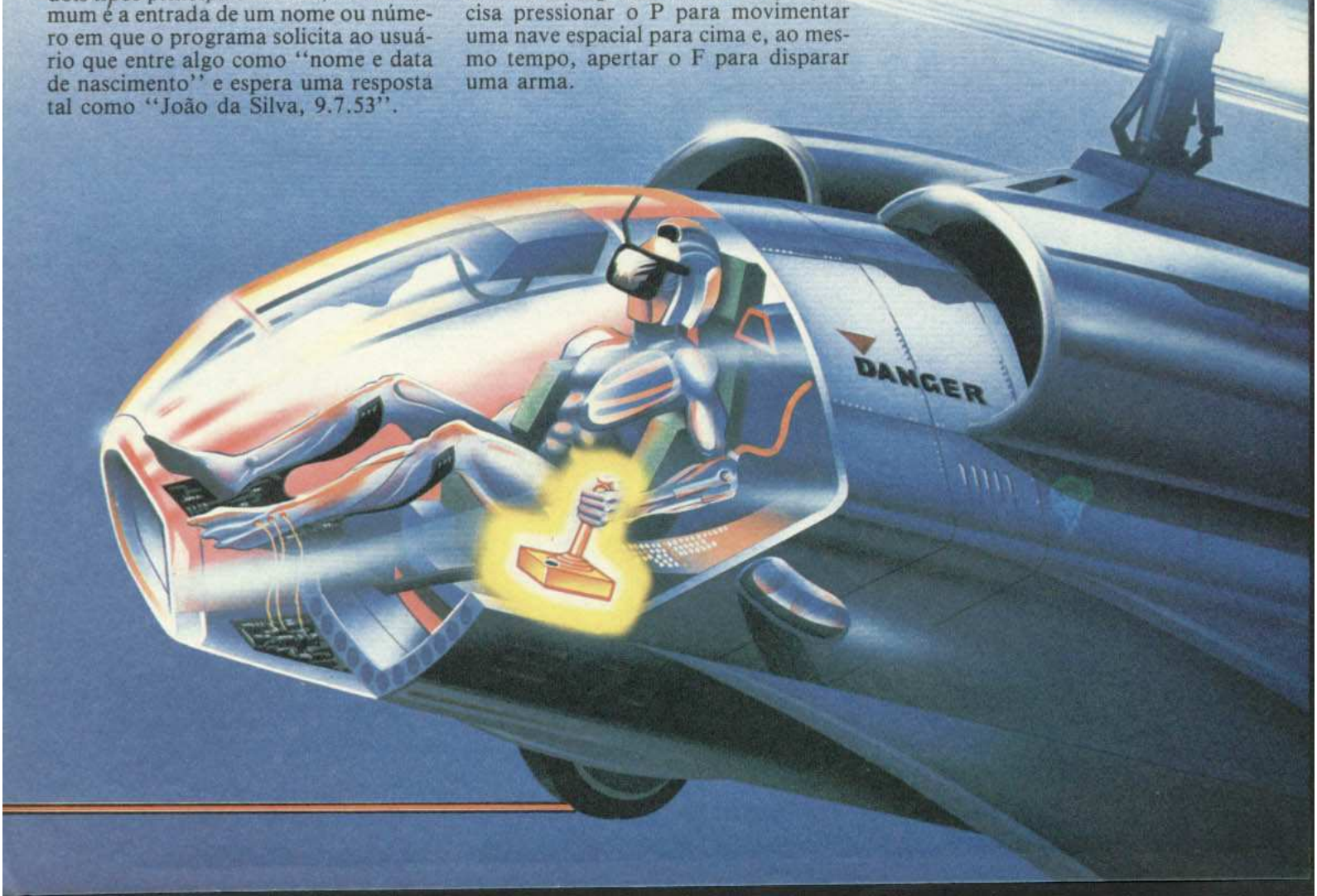
Costuma-se classificar as entradas efetuadas por intermédio do teclado em dois tipos principais. Destes, o mais comum é a entrada de um nome ou número em que o programa solicita ao usuário que entre algo como "nome e data de nascimento" e espera uma resposta tal como "João da Silva, 9.7.53".

O segundo tipo de entrada, por sua vez, transmite um tipo inteiramente diverso de informação. Nela, a pressão a uma tecla — em vez de provocar a entrada do caractere ao qual ela é normalmente ligada — aciona uma outra função bem diferente, determinada pelo programa. Por exemplo, se a tecla X for pressionada, o cursor da tela será movido para a direita; se for Z a tecla pressionada, o cursor será deslocado para a esquerda. Diversos capítulos do curso de *BASIC* e de *Programação de jogos* foram destinados a ensinar a programar esse tipo de entrada.

Obviamente, o primeiro tipo de entrada depende do teclado de modo direto. No segundo tipo, porém, essa ligação não aparece de forma tão evidente. É difícil imaginar, por exemplo, que um usuário pouco familiarizado com o teclado consiga se lembrar de que precisa pressionar o P para movimentar uma nave espacial para cima e, ao mesmo tempo, apertar o F para disparar uma arma.

Felizmente, existe uma alternativa: o joystick, um periférico barato e acessível para microcomputadores. Talvez injustamente associado apenas aos jogos, esse periférico cumpre diversas outras funções.

O joystick é uma espécie de alavanca que pode ser movida em várias direções pelo usuário. A cada movimentação, ele envia ao computador um sinal que indica a sua nova posição. A maioria dos joysticks tem um ou mais botões pulsantes (chamados de *botões de disparo*) que, ao serem pressionados, geram sinais diferentes para o computador (joystick é um termo derivado do aeromodelismo e não tem tradução adequada para o português; a mais aproxi-

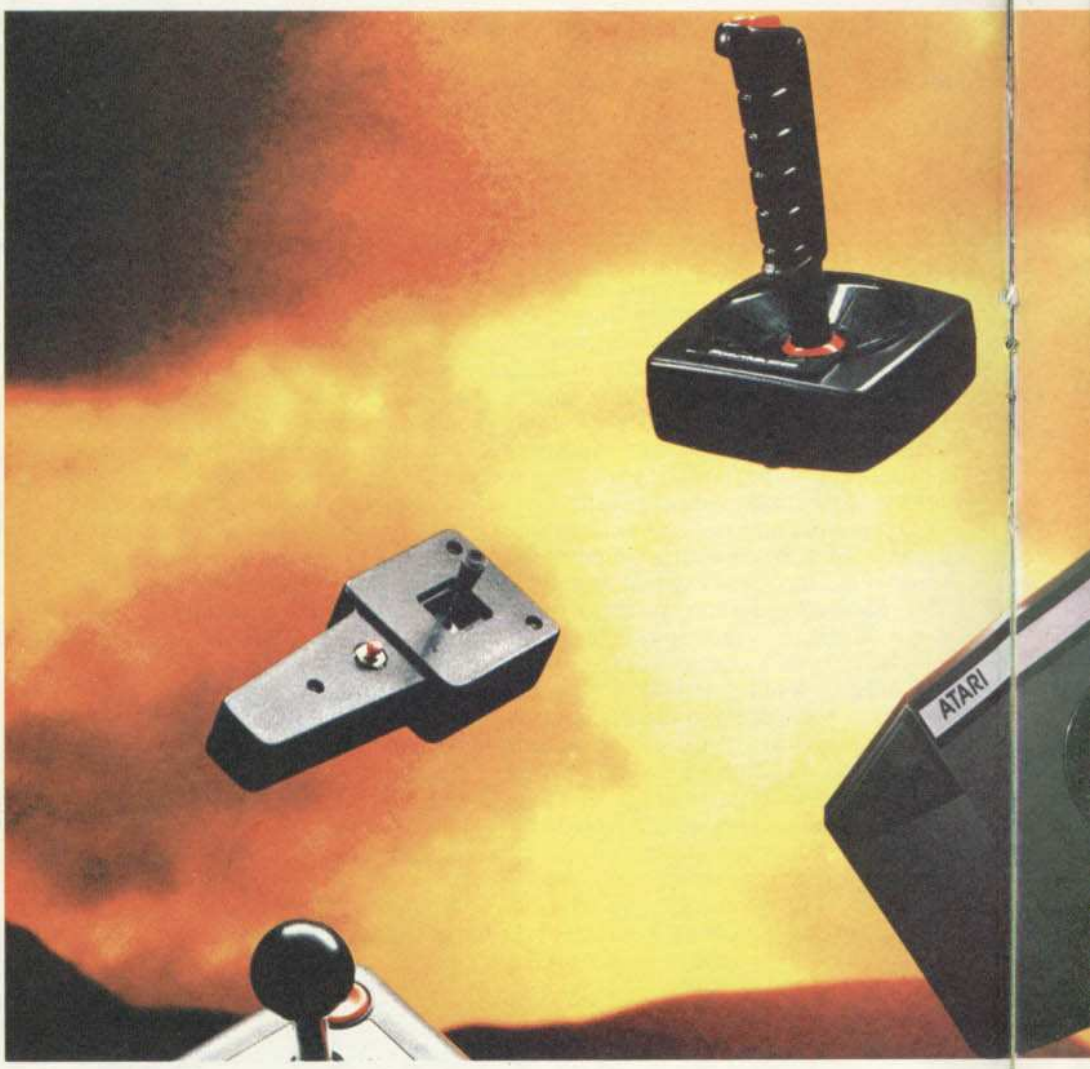


mada seria *manche*).

Com uma programação apropriada, mesmo nos jogos de ação mais modestos, os joysticks podem assumir várias das funções reservadas tradicionalmente para o teclado; assim, eles encontram muitas aplicações interessantes, úteis sobretudo para usuários que não querem ou têm dificuldades em usar o teclado.

Por exemplo, em um programa de gráficos, é perfeitamente possível utilizar um joystick ou um controle semelhante para guiar o cursor que desenhará na tela (na página 164 apresentamos um programa que faz isso por meio do teclado), ou para selecionar uma cor em uma "paleta" exibida na tela. Um joystick também pode ser usado para fornecer respostas alfabéticas ou numéricas. Alguns jogos de fliperama, por exemplo, possuem uma "lista de campeões", na qual são exibidos os nomes dos jogadores e seus recordes de pontos. Neste caso, uma entrada alfabética curta (por exemplo, as iniciais do jogador) é realizada, usando-se o joystick para mover o cursor até a posição onde está a letra desejada e, em seguida, o botão de disparo para selecioná-la. Com um pouco de imaginação, é fácil incorporar esse truque na digitação de nomes e números maiores ou, mais comumente, para efetuar uma escolha entre um menu de opções de programa, por exemplo.

Em um artigo futuro, você verá como programar o seu computador para operar sob o controle de um joystick, de modo a tornar um programa mais atraente ou um videogame mais divertido. Antes disso, porém, é necessário entender quais são os diferentes tipos de joysticks existentes para computadores, e como podem ser utilizados.



TIPOS DE JOYSTICKS

Nem todos os joysticks são adequados aos vários tipos de computadores pessoais. Antes de adquiri-los, convém certificar-se de que eles são compatíveis com a sua máquina. Existe ainda uma outra restrição: ao se comprar um programa de jogos (ou outro qualquer) que utilize joysticks, é necessário verificar cuidadosamente se ele foi escrito para o tipo de joystick que você possui. Dentro dessas limitações básicas podem existir, contudo, diversas opções de compra.

O tipo mais simples de joystick consiste em uma caixa rasa com uma alavanca, que pode ser movimentada em oito direções diferentes: para a esquerda, para a direita, para cima, para baixo e para as quatro posições intermediárias, em diagonal. Além disso, ele conta com um botão de disparo que, apesar do nome, pode ser utilizado para tarefas como selecionar uma letra ou assinalar uma opção. Esse gênero de joystick é muito usado em videogames; por isso, é comumente chamado de "tipo Atari" ou digital.

Alguns desses joysticks possuem dois ou mais botões de disparo que podem ser úteis para jogadores canhotos, por exemplo, ou para aumentar a velocidade ou o grau de conforto na manipulação de jogos difíceis. Esses botões se localizam normalmente na base, mas podem situar-se também na ponta da alavanca de controle. Dependendo do programa, um botão pode, por exemplo, ser usado para disparar um canhão, enquanto o outro aciona um dispositivo que deixa cair uma bomba; um botão pode apagar uma linha de texto, enquanto o outro a edita, e assim por diante. Alguns modelos são especialmente planejados para se amoldarem à mão. Outros são modelados no formato de um cabo de pistola, complementado, em alguns casos, com um gatilho.

Joysticks mais modernos incluem dispositivos extremamente sofisticados, como ponteiros munidos de indicadores ultra-sensíveis de gravidade. Esses indicadores servem para detectar pequenos movimentos ou inclinações da mão do usuário (por exemplo, quando este aponta para alguma locação na tela). Eles utilizam interruptores de mercúrio: sempre que a alavanca é inclinada, o mercúrio em seu interior se desloca para uma das extremidades, fechando ou abrindo um contato elétrico. Tais dispositivos são tão sensíveis que não devem ser usados em jogos, a menos que o programa tenha sido escrito especialmente para eles.

Outro elemento da família dos joysticks é a "raquete eletrônica" (*paddle*), que nada mais é do que um botão giratório, tipo potenciômetro. Neste caso, a posição angular do potenciômetro é detectada pelo computador, que pode usar a informação para mover um cursor na tela, geralmente no sentido horizontal.

Alguns tipos de joysticks resultam de uma combinação de dois paddles, um na posição horizontal e outro na vertical.

Uma outra variante do joystick é a "trackerball" (esfera de comando). Originalmente desenvolvida para aplicações militares e "aviônicas", a esfera de comando foi logo adaptada para máquinas comerciais de jogos de fliperama e estão entrando agora no mercado dos computadores domésticos. Nesse periférico, o controle é feito por meio de uma bola que se projeta acima da superfície da caixa e que pode ser rolada em qualquer direção, com as pontas dos dedos.

Os modelos mais sofisticados desses dispositivos de controle são empregados também em computadores de aplicação comercial e profissional. Um dos mais recentes é o chamado "camundongo" (*mouse*), que se parece com uma trackerball de cabeça para baixo, embora existam também versões que funcionam de acordo com outros sistemas. O camundongo entra em ação quando se rola o dispositivo sobre a superfície de uma mesa. Seus movimentos são detectados por um programa especial, provocando deslocamentos correspondentes de um cursor gráfico ou de texto na tela. Assim, o camundongo pode substituir as teclas de controle do cursor. Ele permite movimentos rápidos sobre a tela para alterar dados ou para solucionar itens de um menu. As seleções são feitas por meio de um ou dois botões no alto da caixa. Ao contrário dos joysticks normais, o camundongo é utilizado para desenhar na tela, mas quase nunca para controlar movimentos em jogos.

Existem muitos outros tipos de dispositivos que servem para controlar o cursor na tela, mas cujo funcionamento é bem diferente do dos joysticks e de seus "parentes". Um deles é o *tablete gráfico*, que consiste em uma mesa retangular, sobre a qual se pode "escrever" com uma caneta especial, ou mesmo com o dedo (*touchpad*). Sensores localizados na superfície do tablete ou na pena transmitem ao computador as coordenadas X e Y da ponta da caneta.

As *canetas ópticas* são outro "parente" dos joysticks, que permitem o desenho direto sobre a tela, a escolha de opções, etc., e estão se tornando cada vez mais populares entre os usuários de computadores domésticos.

COMO FUNCIONAM

Embora pareçam diferentes à primeira vista, na verdade todos esses dispositivos funcionam de maneira semelhante. Em todos eles, o movimento é transformado em uma série de sinais elétricos que são "lidos" pelo computador.

Assim. como qualquer dispositivo eletrônico, os joysticks podem ser tanto digitais quanto analógicos. Normalmente, uma linha de computadores aceita apenas um dos tipos e mais raramente os dois. No joystick digital existem diversos comutadores eletrônicos que são abertos ou fechados conforme o ângulo de movimentação da alavanca: assim, um padrão único de bits é gerado para cada posição. Os paddles e joysticks de tipo analógico, por sua vez, funcionam de forma diferente: eles são construídos com um ou dois potenciômetros (resistores variáveis), cujo ângulo de rotação é proporcional ao deslocamento da alavanca nos sentidos horizontal e vertical. As variações de resistência obtidas com essas rotações são transmitidas ao computador na forma de duas voltagens proporcionais, de modo a fornecer uma única combinação para cada posição de controle.

Do ponto de vista visual, os dois tipos de joystick são também bastante diferentes. No tipo analogico, os potenciômetros são montados em ângulos retos e operados por um enlace mecânico. A alavanca, em conseqüência, não fica balanceada ao centro, ou seja, permanece na posição em que o usuário a deixar. Os tipos digitais, ao contrário, são normalmente balanceados ao centro e, quando liberados, voltam automaticamente para a posição central ou neutra. No entanto, alguns modelos analógicos são balanceados ao centro.

Por outro lado, uma nítida diferença no comportamento mecânico separa os dois sistemas. Os joysticks digitais (balanceados ao centro) são mais duros e normalmente só se movimentam por uma pequena distância. Assim, um esforço maior feito pelo jogador é contrabalançado por uma maior resistência mecânica.

Na verdade, o efeito que ambos os ti-

pos provocam no computador é controlado mais pelo programa do que pelo próprio joystick. Geralmente, um programa bem escrito torna o joystick mais sensível e fácil de controlar. Por exemplo, um programa pode determinar que um leve movimento da alavanca seja suficiente para dar início a mudanças de direção de um objeto na tela, sem que haja necessidade de se pressionar continuamente a alavanca.

Já os tabletes digitalizadores que funcionam por toque (*touchpads*) consistem em um sanduíche de duas folhas de plástico, separadas por uma pequena distância. Uma fina grade de resistores ou fios condutores é construída sobre uma das faces dessas folhas, orientadas de modo a formar ângulos retos entre si. Ao menor contato de um dedo ou de um lápis com o tablete, os condutores situados nas duas folhas se tocam e geram um padrão de voltagens, que é varrido pelo computador. Os touchpads, entretanto, podem apresentar problemas, devido à dificuldade de se conseguir uma superfície uniformemente sensível. Uma desvantagem em utilizá-los é que, se a superfície se sujar — por marcas de dedos, por exemplo —, os movimentos tenderão a se tornar erráticos devido ao deslizamento do dedo pela superfície.

Extremamente leves, as trackerballs podem ser operadas com facilidade, pois a bola não tem conexão mecânica direta com o sistema sensor. Ela é sustentada sobre rolamentos que giram livremente em duas direções, em ângulo reto uma em relação à outra. Quando a bola sofre uma rotação, ela faz rodar um ou ambos os rolamentos. Estes, por sua vez, estão conectados a potenciômetros ou a um sensor digital — um sistema formado por um disco rotatório para interromper um raio de luz proveniente de um diodo LED (foto-emissor), que cai sobre um fototransistor (foto-sensor). Com a contagem de impulsos de interrupção, o computador fica "sabendo" de quanto girou o rolamento. Mais uma vez, qualquer que seja o sistema utilizado, o computador é capaz de interpretar os sinais elétricos em termos de um padrão determinado de movimentação na tela.

OPERAÇÃO COM O COMPUTADOR

Os comandos **GET$** ou **INKEY$** do BASIC são os mais utilizados para programar a execução de funções de movimentação do cursor de vídeo, sob controle de determinadas teclas do teclado. Eles efetuam uma "varredura" do teclado, de modo a assinalar ao programa se alguma tecla foi pressionada. Se isso acontecer, o programa poderá efetuar alguma operação específica — tal como a movimentação de uma base de mísseis para a direita, se a tecla X, por exemplo, for pressionada.

A operação do joystick digital não difere essencialmente disso. Mas enquanto o teclado é parte integrante do computador, o joystick é um elemento externo acoplado a ele. Assim, o joystick deve primeiro ser conectado ao computador através de uma porta adequada (um conector especial); é por essa porta que o micro normalmente se comunica com o mundo exterior. Em seguida, é necessário colocar na memória RAM do micro um programa adequado para "varrer" periodicamente a porta de entrada, de modo a procurar por um sinal específico, que signifique que o joystick está sendo movimentado.

Alguns problemas de compatibilidade podem surgir entre esses periféricos e o computador. Em primeiro lugar, o joystick precisa ser conectável à porta de entrada do micro, seja diretamente, seja através de algum tipo de interface. Além disso é necessário que o programador conheça os sinais gerados pelo joystick; caso contrário, não será possível programar o computador para procurá-los.

Essas exigências dão lugar, às vezes, a tremendas confusões. Felizmente, já existem padrões de mercado seguidos em maior ou menor grau por diferentes fabricantes. No Brasil, cada linha de microcomputadores apresenta um conjunto bem definido de convenções. O padrão mais comum, que se tornou virtualmente universal, é o do joystick tipo Atari (digital e centro-balanceado). Existem diversos fabricantes de joysticks desse tipo, e alguns computadores são projetados de modo a aceitá-los diretamente.

É o caso, por exemplo, da linha Sinclair (ZX-81 e Spectrum), cujos compatíveis nacionais (TK-85, TK-90X, etc.) incluem um conector para joysticks tipo Atari. Os modelos da linha MSX têm, por sua vez, entrada para dois joysticks tipo Atari. Os compatíveis com o TRS-Color (por exemplo, o Prológica CP-400) utilizam joysticks analógicos. Já os micros da linha TRS-80 normalmente não contam com nenhum tipo de entrada para joystick.

Além disso, podem existir, às vezes, diferenças importantes entre diversos modelos de máquinas da mesma linha, como no caso do Apple, que tem protótipos nacionais que aceitam joysticks digitais tipo Atari (é o caso do TK-2000), ou analógicos (*paddles*).

Uma das diferenças mais importantes entre micros diz respeito ao interfaceamento.

INTERFACEAMENTO

Como qualquer periférico de computador, os joysticks precisam ser conectados à UCP por intermédio de uma interface adequada. A interface pode estar integrada à configuração básica da UCP do computador, ou pode ser um dispositivo separado.

Muitos fabricantes incorporam aos seus micros um conector e uma interface do tipo Atari, de modo a fazê-los aceitar qualquer joystick que seja de um modelo compatível com esse padrão.

Outra saída seria dispor de uma ou mais portas analógicas, que aceitam paddles ou joysticks do tipo potenciômetro. No entanto, são raros os micros que já incluem essas portas em sua configuração básica, pois isto significa que devem ser construídos dois conversores analógico-digitais para cada joystick a ser utilizado.

S

Os micros da linha Sinclair Spectrum não vêm acompanhados de joysticks. Assim, para ajustar a eles um desses periféricos, é necessário uma interface apropriada ao conector de borda existente atrás do console.

A interface é alojada em uma caixa separada que se situa na parte posterior da máquina, em contato direto com os terminais do conector de borda, na entrada do usuário. Existe, porém, um tipo de interface (disponível apenas no Exterior) construída dentro da caixa do próprio joystick, com um conector separado para o cabo de ligação.

O joystick mais adequado para o Spectrum é compatível com o tipo Atari. No Exterior, entretanto, existem muitos outros tipos de joysticks, que são incompatíveis entre si no que se refere aos sinais que geram no conector de expansão.

Por isso, o usuário do Spectrum deve tomar muito cuidado quando comprar ou copiar programas provenientes do Exterior que utilizem joysticks, pois eles nem sempre funcionam com o joystick disponível no Brasil. Assim, é conveniente verificar a documentação do software antes de adquiri-lo. Alguns programas mais sofisticados incluem um menu de opções que permitem ao usuário selecionar o tipo de joystick a ser usado para acionar o jogo.

Outro problema para os proprietários

do Spectrum é que as interfaces mais comuns desse micro só possibilitam a conexão de um joystick. Entretanto, algumas interfaces mais sofisticadas permitem a conexão de dois joysticks ao mesmo tempo. Infelizmente, porém, são raros os softwares compatíveis com tais interfaces.

O interpretador BASIC do Spectrum não tem nenhum comando específico para programação com joysticks. Assim, o mais comum é usar o comando **INKEY$**.

O ZX-81 está longe de ser uma máquina ideal para jogos. Entretanto, assim como o Spectrum, é possível conectá-lo a um joystick do tipo Atari por intermédio de um conector Phillips, situado na lateral ou na traseira do console. Portanto, a operação com esse micro é parecida com as descritas acima.

O BASIC do ZX-81 não tem nenhum comando específico para a programação de joysticks, como é o caso dos micros da linha MSX, Apple e TRS-Color, mas o comando **INKEY$** pode ser usado da mesma forma que com o teclado, pois os joysticks para o ZX-81 geram sinais correspondentes às teclas de controle de cursor (6, 7, 8 e 9, combinadas com <**SHIFT**>).

Isto é uma vantagem adicional, pois um jogo que utilize tais teclas operará igualmente bem, com ou sem joystick (só que ficará mais divertido com o joystick, como é o caso de um popular programa de simulação de vôo de uma aeronave).

Os microcomputadores da linha TRS-Color incluem duas entradas para joysticks analógicos, do tipo potenciômetro de centro não-balanceado. Esses joysticks têm botões de disparo, e podem ser facilmente programados por comandos específicos do interpretador BASIC. A tela de alta resolução gráfica, quando associada por programação ao uso desses joysticks, permite efeitos sensacionais de animação gráfica, em jogos ou no desenho livre.

Os micros da linha TRS-80 não incluem nenhuma previsão ou porta especial para joysticks.

Entretanto, esses computadores podem ser conectados tanto a joysticks digitais do tipo Atari, quanto a joysticks e paddles analógicos; para isso, existem produtos comercialmente disponíveis no mercado.

No primeiro caso, é necessário efetuar uma modificação no hardware, conectando fios em paralelo com a interface do teclado. Em seguida, deve-se ligar a máquina ao joystick. Uma vez em operação, este gera sinais semelhantes aos provocados pela pressão das teclas de controle do cursor, que podem ser detectadas por meio do comando **INKEY$**, do BASIC. Há um dispositivo fabricado no Brasil — "joypad", um pequeno teclado paralelo apenas com as teclas de controle de cursor (*joypad*) — cuja função consiste em facilitar o desempenho em jogos.

Para conectar dispositivos analógicos, é necessário adquirir uma interface especial de conversão analógico-digital, que é ligada à porta de expansão, na parte de trás da UCP.

No TRS-80 não existem comandos específicos do BASIC para os dois tipos de joysticks.

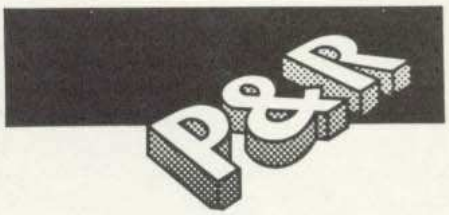
Os microcomputadores da linha MSX incluem portas de entrada para joysticks do tipo digital centro-balanceado, com dois ou mais botões de disparo. No Brasil, cada fabricante oferece o seu modelo próprio de joystick, mas o padrão de conexão e geração de sinais segue o do Atari.

Da mesma forma que o TRS-Color, o interpretador BASIC dessas máquinas contém comandos bastante poderosos.

A maioria dos micros da linha Apple apresenta no console duas portas para dois paddles analógicos, ou para um joystick de dois eixos (não centro-balanceado). O interpretador BASIC incorpora comandos para leitura da posição dos potenciômetros e da pressão ao botão de disparo.

Existem ainda no mercado joysticks do tipo digital, que podem ser conectados via porta serial ou paralela.

O Microdigital TK-2000 inclui, na versão padrão, um conector específico para joystick digital do tipo Atari, que opera paralelamente ao teclado normal (ou seja, gera sinais correspondentes às teclas de controle do cursor e às teclas **FIRE** dos dois lados do teclado). Assim, a utilização do joystick pode ser programada por meio de comandos **GET$** ou **PEEK** do teclado existentes no BASIC do TK-2000. Não existem joysticks ou paddles do tipo analógico para esse computador.



Como funciona um conversor analógico-digital?

Necessário para converter em números binários as voltagens contínuas geradas pelos paddles e joysticks analógicos, o conversor A/D realiza uma tarefa de *digitalização* do sinal elétrico de entrada.

Suponhamos que uma rotação de 118 graus num dos potenciômetros do joystick gera uma voltagem proporcional entre 0 e 5 volts. O conversor A/D tem um comparador interno que gera voltagens sucessivas, divididas em um certo número de intervalos (por exemplo, 256 intervalos de 0.0195 volts cada, se for um conversor de oito bits). A cada nível interno de voltagem, o circuito compara-o com o nível externo de voltagem (gerado pelo joystick) e decide se esses valores são aproximadamente iguais ou não. Ao mesmo tempo, outro circuito digital vai contando quantos desses níveis foram testados. Assim que o comparador informar que a igualdade foi atingida, o conversor A/D enviará ao computador, através da interface paralela, o byte contendo o número de contagens. Por exemplo, se a voltagem gerada pelo joystick for 1.95 volts, o número enviado para o computador será 100!

CUIDADO AO COMPRAR UM JOYSTICK!

Ao comprar um joystick, verifique primeiro se o modelo escolhido é compatível com o seu computador e, no caso do Spectrum, com o seu software também.

Em seguida, pense em quanto você está disposto a gastar; se você é proprietário de um Spectrum, tenha em mente que a interface poderá custar tanto quanto o joystick, ou até mais. Os tipos mais simples de joysticks custam o mesmo que um programa de jogo, enquanto os mais elaborados podem ser duas ou três vezes mais dispendiosos.

ORDENAÇÃO PELO MÉTODO DE BOLHAS

Uma vez elaborado o projeto geral de um programa e escritos os módulos individuais, pode-se começar a pensar em testar os módulos. Em seguida, é necessário planejar a forma de colocá-los todos juntos.

As sub-rotinas ou módulos precisam ser colocados de alguma forma dentro do contexto do programa. Normalmente, a ligação é feita por meio de variáveis. Algumas destas — conhecidas como *parâmetros de entrada* — são especificadas no início, e passadas à rotina. Outras variáveis retornam ao programa por intermédio da rotina; estas são os *parâmetros de saída*. É muito importante que as variáveis sejam especificadas de um modo preciso para não serem confundidas umas com as outras.

Ao se começar a escrever um programa deve-se fazer uma lista de todas as variáveis necessárias, juntamente com uma descrição dos seus usos e possíveis valores iniciais, caso sejam conhecidos. Do contrário, mesmo que se saiba no início o que significam todas as letras, corre-se o risco de esquecer de retornar ao programa mais tarde. Um recurso bastante prático, caso o computador permita ou o espaço de memória não seja muito pequeno, consiste em adotar nomes longos para as variáveis (veja na página 99 uma tabela do que é permitido para cada linha de microcomputadores).

ROTINA DE ORDENAÇÃO TIPO BOLHA

Como se deve especificar as variáveis para uma rotina de ordenação tipo bolha? O exemplo a seguir apresenta uma porção específica de matriz ordenada alfabeticamente.

Variáveis de entrada:
A$(N) — conjunto unidimensional a ser classificado (tamanho de **N > = 1**)
N1 — primeiro item em conjunto a ser classificado (**1 < = N1 < = N2**)
N2 — último item em conjunto a ser classificado (**N1 < = N2 < =** o tamanho de **A**)

Variáveis de saída:
A$(N) — conjunto ordenado.

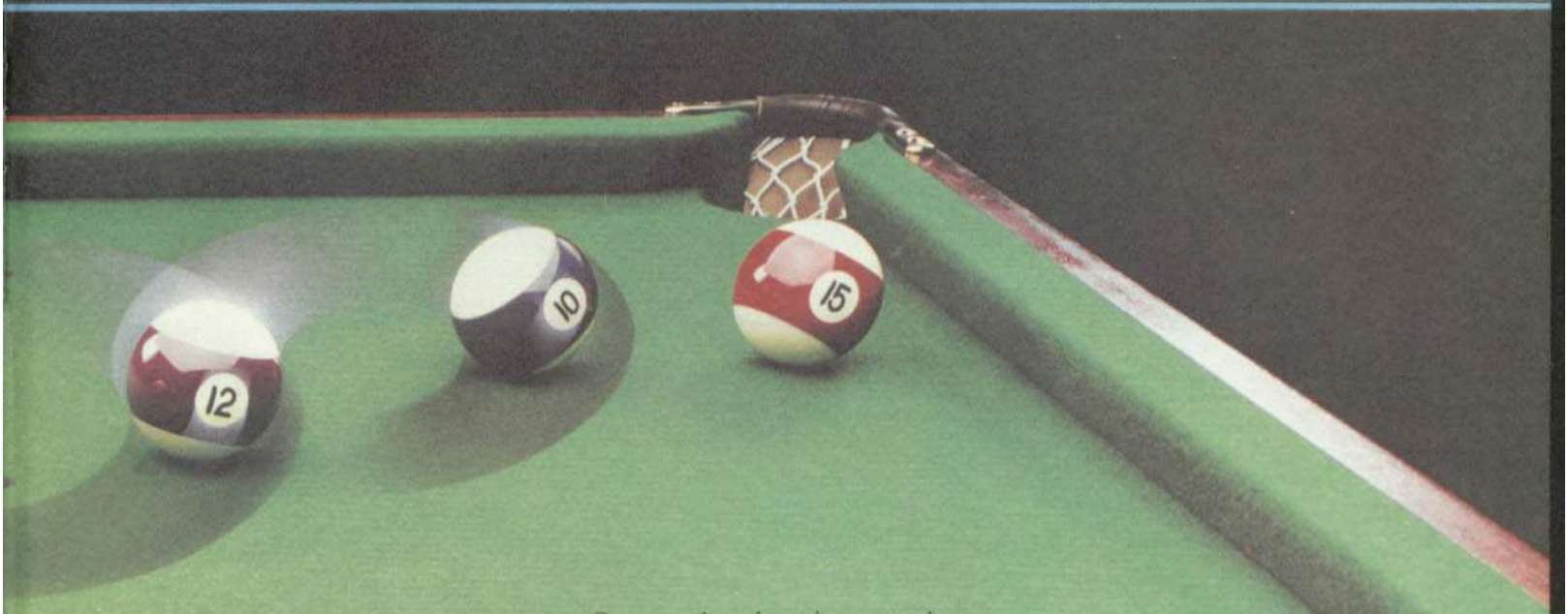
Variáveis temporárias:
Z, Z$, I.

Uma providência muito útil para evitar conflitos entre módulos ou com o resto do programa consiste em listar as variáveis temporárias utilizadas em uma sub-rotina. Também é útil reservar algumas letras, especialmente para variáveis temporárias: por exemplo, as variáveis de **Z0** a **Z9**. Esse tipo de recurso evita desperdícios de espaço das variáveis.

Alguns erros ou defeitos de programa são causados às vezes por variáveis *viciadas* — isto é, com valores mudados.

Aperfeiçoe as técnicas da programação estruturada, aprendendo a construir um programa de ordenação pelo método de bolhas, e coloque em ordem o que quiser.

- DEFINA AS VARIÁVEIS
- COMO ESCREVER UMA SUB-ROTINA DE ORDENAÇÃO
- COMO TESTAR OS MÓDULOS
- COLOQUE TUDO JUNTO



Esses problemas, contudo, podem ser evitados com a aplicação do método exposto acima.

Uma listagem de variáveis também é útil para o caso de o programa vir a ser modificado mais tarde. Essa medida assegura maior rapidez na alteração das variáveis, impedindo ao mesmo tempo o aparecimento de erros extras no programa. Tais erros são normalmente provocados por variáveis viciadas, utilizadas para outros propósitos. Lembre-se ainda de anotar as modificações, juntamente com as datas em que foram feitas.

Eis aqui o programa para uma rotina de ordenação por bolhas (seu funcionamento será explicado mais adiante, na lição número 23 de *Programação BASIC*).

```
1000 REM ORDENAÇAO TIPO BOLHA (
A$(N),N1,N2)
1010 LET Z=0
1020 FOR I=N1 TO N2-1
1030 IF A$(I)<=A$(I+1) THEN GOT
O 1080
1040 LET Z$=A$(I)
1050 LET A$(I)=A$(I+1)
1060 LET A$(I+1)=Z$
1070 LET Z=1
1080 NEXT I
1090 IF Z=1 THEN GOTO 1010
1100 RETURN
```

Como toda sub-rotina, esta deve ser chamada a partir do programa principal. Por exemplo, para ordenar os itens de 5 a 20, a rotina pode ser chamada assim:

```
100 LET N1=5:LET N2=20:GOSUB
1000: REM ORDENACAO TIPO BOLHA
```

É preciso também que exista uma seção do programa que lhe permita entrar os itens a serem ordenados, e outra que imprima a lista ordenada. Neste estágio você deve decidir como quer que a exibição apareça na tela.

Se você trabalha com um MSX, pode tirar proveito do poderoso BASIC desse micro. Mude a linha 1040 para **SWAP A$(1),A$(I+1)** e elimine as linhas 1050 e 1060. Seu computador trocará o conteúdo das duas variáveis de uma só vez, dispensando o uso da variável **Z$**.

COMO TESTAR OS MÓDULOS

Cada módulo do projeto original pode se tornar uma sub-rotina no seu programa. Esse método de dividir o programa é muito útil durante o estágio de teste, pois os módulos podem ser postos à prova ou depurados individualmente. Volte agora à sub-rotina da ordenação tipo bolha, testando-a assim:

```
8 INPUT "Número de ítens ";N
10 DIM A$(N)
12 PRINT"Entrada dos ítens da m
atriz:"
14 FOR I=1 TO N: INPUT A$(I):NE
XT I
16 INPUT "Intervalo a ser orden
ado ";N1,N2
18 GOSUB 1000
20 PRINT"Lista ordenada:"
22 FOR I=1 TO N:PRINT A$(I):NEX
T I
24 GOTO 16
```

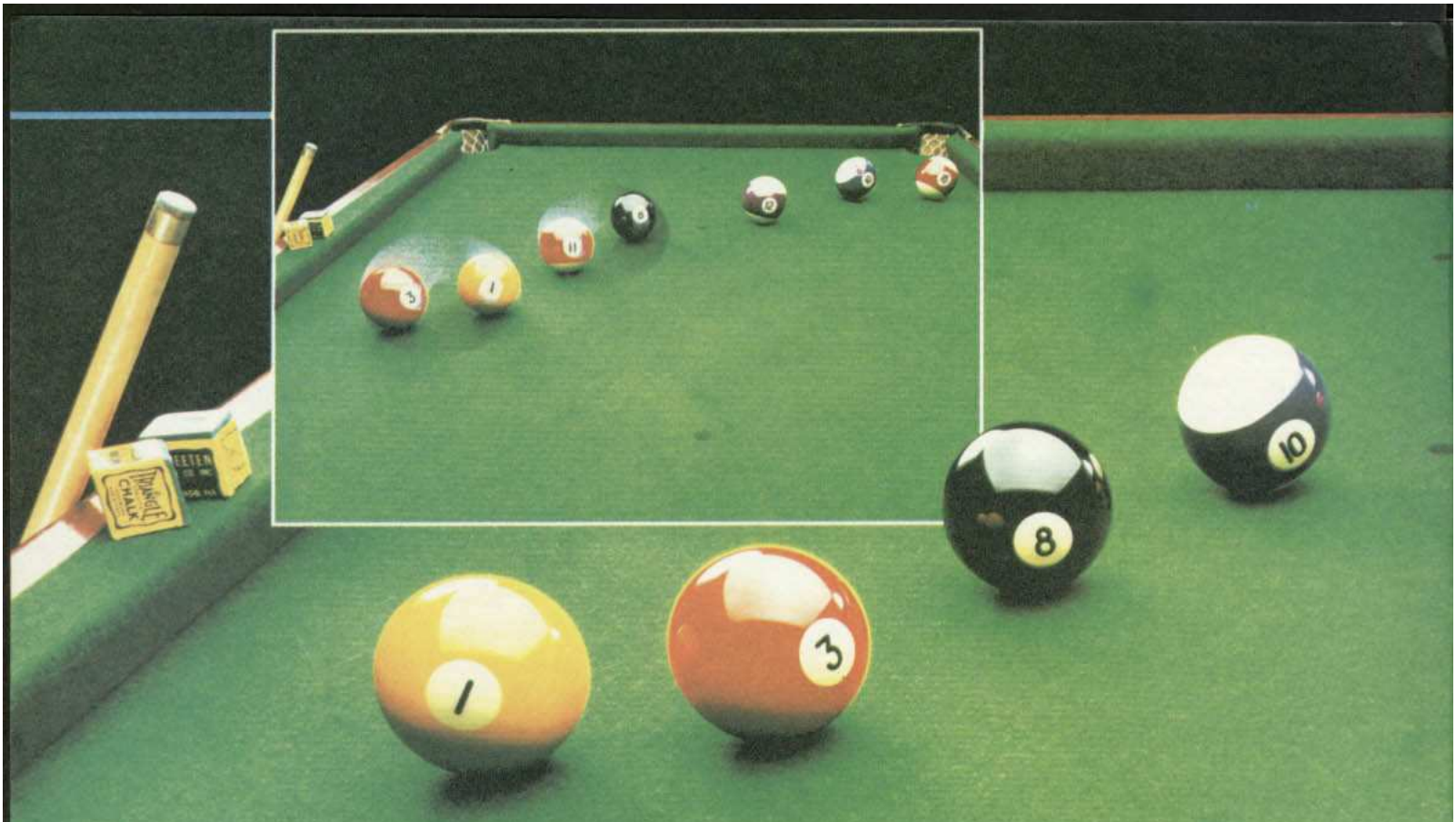
O programa acima também funcionará no TRS-Color e TRS-80, desde que se adicione a linha a seguir, cujo objetivo é reservar um espaço suficiente de memória:

```
6 CLEAR 1000
```

No ZX-81 e no Spectrum, modifique a linha 10 do programa acima para:

```
10 DIM A$(N,10)
```

No ZX-81, fracione as linhas com declarações múltiplas, mude tudo para maiúsculas, e altere a linha 8 para:



```
8 PRINT "NUMERO DE ITENS"
9 INPUT N
```

Entretanto, a tarefa de testar cada caso de entrada e de saída pode se revelar impossível em programas complexos, exigindo um tempo excessivamente grande. Apesar disso, os casos limítrofes podem ser checados. Por exemplo, a rotina seguinte pode ser verificada para entrar valores de 1,99 e 100.

```
1010 PRINT "ENTRE UM NUMERO
     (1-99) ";
1015 INPUT N
1020 IF N<1 OR N>99 THEN GOTO
     1010
1030 RETURN
```

Outra providência aconselhável é certificar-se de que cada linha do programa foi rodada pelo menos uma vez durante o estágio de depuração. Assim, todos os desvios condicionais, tal como a declaração **IF**, poderão ser chocados com ambas as condições de verdadeiro ou falso.

COLOQUE TUDO JUNTO

Finalmente, todos os módulos poderão ser encadeados e o programa testado como um todo. Isso é conhecido como *integração do programa*. Se ocorreram problemas, qualquer módulo suspeito poderá ser checado novamente e modificado em caso de necessidade.

Cumpridas todas essas exigências, você terá um programa perfeitamente estruturado que fará exatamente o que for determinado.

As regras para se escrever um programa estruturado são, resumidamente, as seguintes:

1. Escreva uma descrição geral do programa.
2. Divida-a em tantos módulos quantos forem os níveis necessários.
3. Desenhe um fluxograma para cada módulo e defina as variáveis de entrada e de saída e de quaisquer outros efeitos, tais como a exibição na tela.
4. Escreva os programas para cada módulo que utilizar as estruturas descritas na parte 1.
5. Teste os módulos, fornecendo as entradas e checando os resultados e as saídas.
6. Combine todos os módulos e teste o conjunto do programa.

A finalidade de todo esse trabalho é aumentar a legibilidade de um programa, assim como sua capacidade, segurança e transportabilidade. Além disso, ele facilita a tarefa de testar e modificar o programa.

COMO FUNCIONA O MÉTODO DAS BOLHAS

O programa de ordenação pelo método das bolhas foi utilizado aqui para mostrar como um módulo pode ser construído e depois testado, antes de ser encadeado ao programa principal. As rotinas de ordenação são muito úteis para todos os tipos de programas. A que apresentamos agora classifica palavras em ordem alfabética, mas poderia servir também para classificar números em ordem crescente. Basta modificar as variáveis **Z$** para **Z**, e o conjunto **A$( )** para **A( )**.

O computador percorre a lista, comparando pares de itens, um de cada vez. Se eles estiverem em ordem correta não serão mudados. Se estiverem em ordem incorreta serão intercambiados. O programa continua através da lista, efetuando mais trocas até que todos os itens estejam em ordem correta.

Para avaliar o funcionamento do programa em detalhes é conveniente compará-lo com o fluxograma. A primeira parte do programa (que não está no diagrama) define o número de itens da lista — **N** — e estabelece um conjunto chamado **A$( )** com espaço suficiente para **N** itens. As linhas 12 e 14 pedem ao usuário as palavras que serão armazenadas no conjunto e a linha 16 pergunta quais delas serão ordenadas. Se você quiser ordenar a lista inteira, digite 1 seguido de uma vírgula, mais o valor de **N**. A sub-rotina é chamada na linha 18.

A rotina começa estabelecendo Z igual a 0. Z é conhecido como um *sina-*

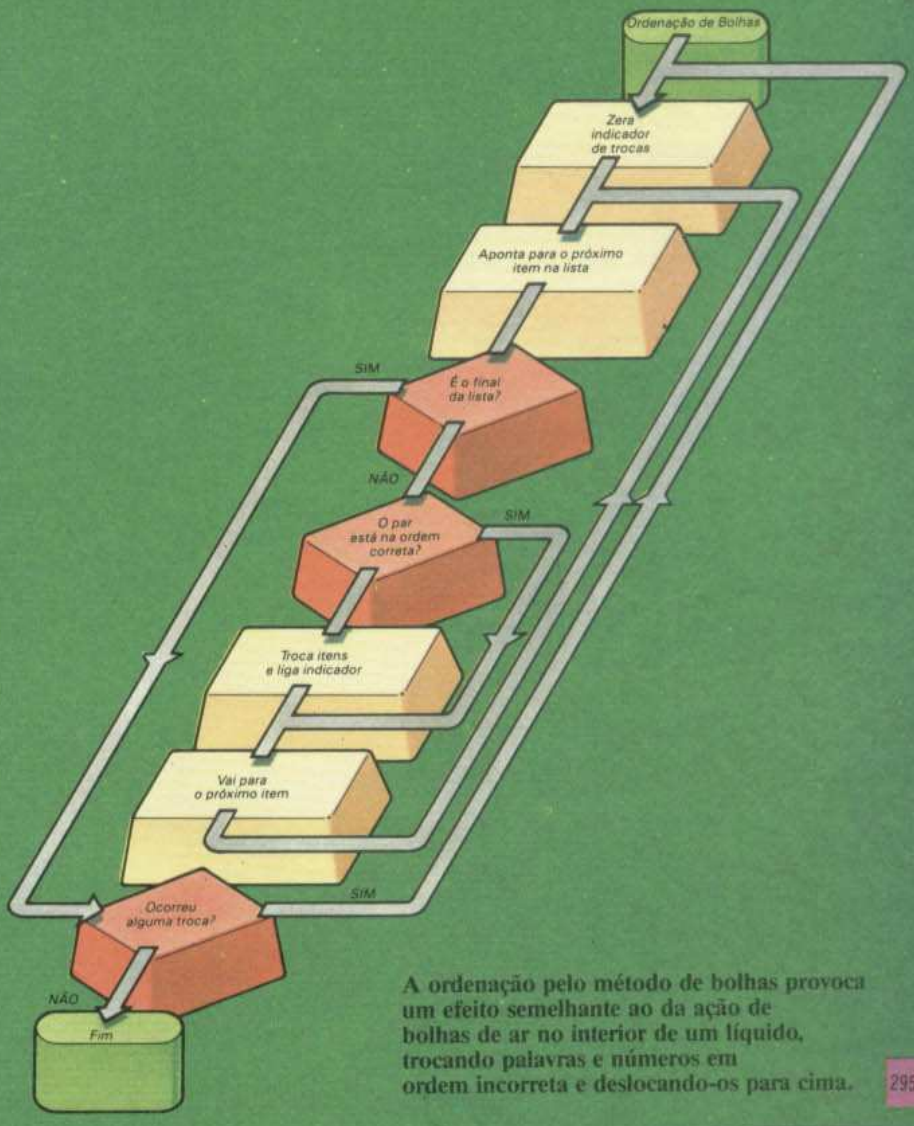
lizador e registra se alguma troca de posições foi realizada.

A linha 1020 cria um laço para percorrer a lista. Os números certificam de que cada par foi comparado pelo menos uma vez. A linha 1030 compara as primeiras duas palavras; se estas estiverem em ordem correta, o programa pulará a rotina de trocas e se dirigirá para o par seguinte.

As linhas 1040 a 1070 somente serão atingidas se as palavras estiverem em ordem incorreta. A primeira palavra é colocada em uma variável temporária **Z$**. A segunda é movimentada para alguma posição, e então a primeira palavra é deslocada uma posição abaixo, dentro do conjunto. Z é então igualado a 1 para mostrar que foi efetuada uma troca. A linha 1080 manda o computador de volta para comparar o par de palavras seguinte.

Uma vez que todos os pares tenham sido comparados, o programa atingirá a linha 1090. Se Z = 1, isso significa que pelo menos uma troca foi efetuada; dessa forma, ele percorrerá a lista novamente. Se nenhuma troca foi efetuada, então a lista está na ordem correta, a sub-rotina termina e as linhas 20 e 22 imprimirão a lista ordenada.

A denominação de método das bolhas tem, assim, sua razão de ser: o movimento das palavras lembra globos de ar em ascensão no interior de um líquido.



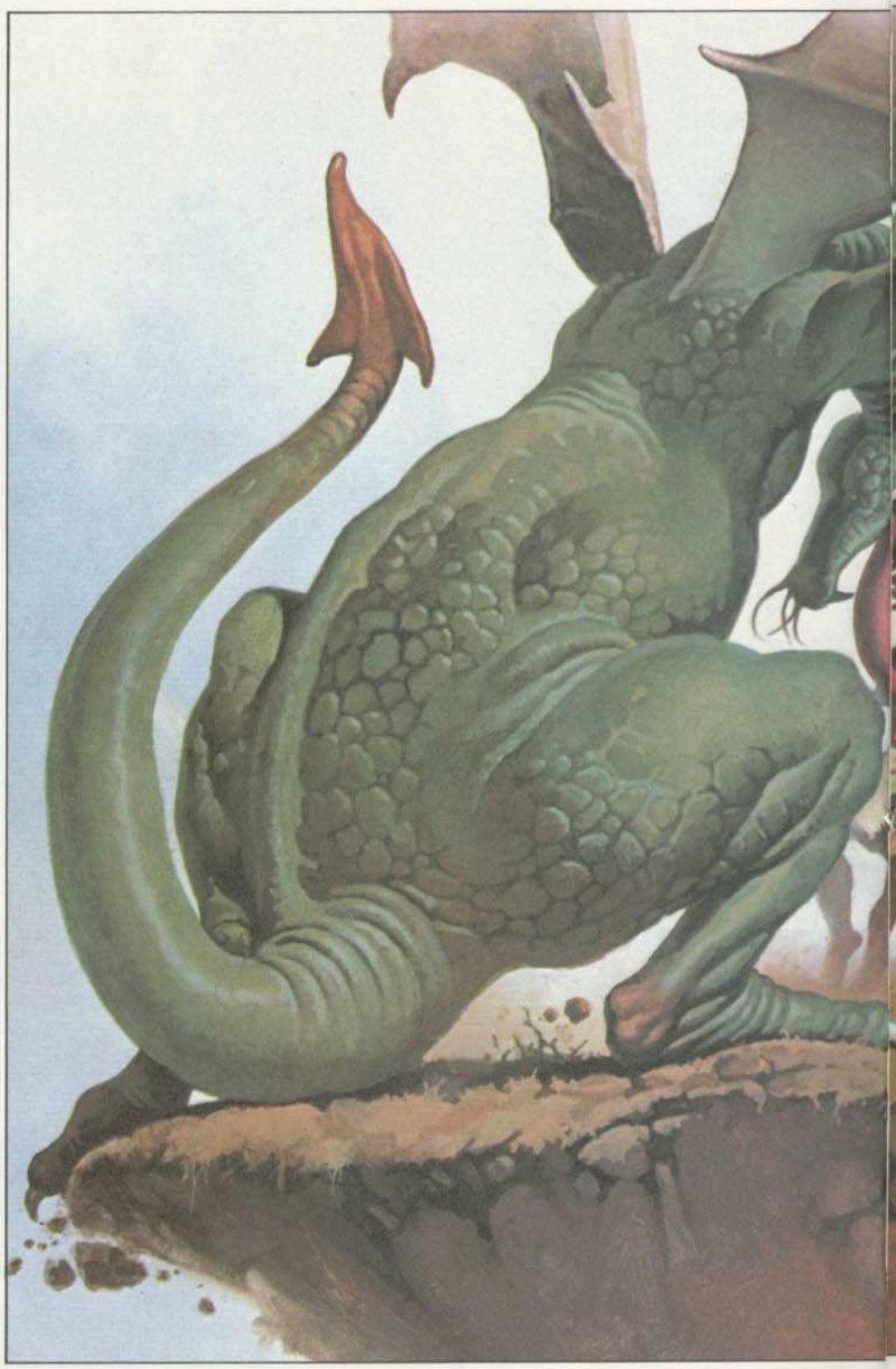
A ordenação pelo método de bolhas provoca um efeito semelhante ao da ação de bolhas de ar no interior de um líquido, trocando palavras e números em ordem incorreta e deslocando-os para cima.

ASSEMBLER PARA O TRS-COLOR

Bem mais rápido do que o programa para o Spectrum, o Assembler para os micros da linha TRS-Color proporciona igual eficiência. Economize tempo, aprendendo a trabalhar com ele.

Nesta lição, apresentamos um Assembler para o TRS-Color. Mais longo do que o programa para o Sinclair ZX Spectrum objeto da lição anterior, ele conta com a participação de um editor.

O microprocessador utilizado pelo TRS-Color é o Motorola 6809 de oito bits, cujo conjunto de instruções é bem menor que o do Z80, empregado pelo Spectrum. Entretanto, o Assembler do TRS-Color tem que passar por até três "etapas" para traduzir um comando de dezesseis bits, enquanto o programa do Spectrum deve passar por no máximo duas. Apesar dessa etapa adicional, o programa do TRS-Color é quase três vezes mais rápido que o do Spectrum: seu grande recurso para procurar os mnemônicos — a função **INSTR** — não existe no BASIC do Spectrum.



A linha 10 pode provocar um erro FC quando o programa é rodado pela primeira vez. Não se importe com a mensagem de erro e rode o programa novamente, que ele funcionará.

O ASSEMBLER

```
10 PMODE 0:PCLEAR 1:CLEAR 3000:
CLS:PRINT @233,"INICIALIZANDO":
R$=CHR$(13):POKE 146,1
20 DIM SK$(1),K1(94),K2(94),T$(
200),RR(100),Z$(100)
30 FOR CC=1 TO 94:READ K$,K1(CC
),K2(CC):C=CC/49:SK$(C)=SK$(C)+
RIGHT$(STR$(CC),2)+K$:NEXT
40 DATA ADCA,185,1,ADDA,187,1,A
DDD,243,2,ASL,120,3,CLR,127,3,C
MPA,177,1,CMPD,4275,2,CMPY,4284
,2,BCC,36,4,BCS,37,4,BEQ,39,4
50 DATA BHS,36,4,BLO,37,4,BMI,4
3,4,BNE,38,4,BPL,42,4,BRA,32,4,
LBRA,22,5,BSR,141,4,LBSR,23,5,C
MPX,188,2,CMPU,4531,2,CMPS,4540
,2,DEC,122,3
60 DATA INC,124,3,JSR,189,3,LDA
,182,1,LDB,246,1,LDD,252,2,LDS,
4350,3,LDU,254,3,LDX,190,3,LDY,
4286,3,LSL,120,3,LSR,116,3
70 DATA PSHS,52,1,PSHU,54,1,PUL
S,53,1,PULU,55,1,ROL,121,3,ROR,
118,3,RTS,57,,STA,183,3,STB,247
,3,STD,253,3,STS,4351,3,STU,255
```

- CONVERSÃO AUTOMÁTICA DA LINGUAGEM ASSEMBLY PARA CÓDIGO DE MÁQUINA
- COMO CALCULAR SALTOS E DESVIOS
- MANIPULAÇÃO DE PÓS-BYTES
- USO DE RÓTULOS
- COMO COLOCAR CÓDIGO DE MÁQUINA NA MEMÓRIA
- AUMENTA A VELOCIDADE


```
,3
80 DATA STX,191,3,STY,4287,3,SU
BA,176,1,SUBD,179,1,ANDA,180,1,
ABX,58,,ANDCC,76,1,ASR,119,3,RT
I,59,,SBCA,178,1,NOP,18,,NEG,11
2,3
90 DATA BITA,181,1,BGE,44,4,BGT
,46,4,BHI,34,4,BLE,47,4,BLS,35,
4,BLT,45,4,BRN,33,4,BVC,40,4,BV
S,41,4,EXG,30,1,TFR,31,1
100 DATA COM,115,3,CWAI,108,1,D
AA,25,,EORA,184,1,TST,125,3,LEA
S,66,3,LEAU,67,3,LEAX,64,3,LEAY
,65,3,MUL,61,,ORA,138,1,ORB,202
,1
110 DATA ORCC,74,1,SEX,29,,SWI,
63,,SWI2,4159,,SWI3,4415,,SYNC,
19,,EQU,-1,2,FCB,-2,1,FDB,-3,2,
RMB,-4,,JMP,126,3
120 DIM X$(13),V(14),KK(13),Y$(
13)
130 FOR C=0 TO 12:READ X$(C),V(
C),KK(C),Y$(C):NEXT
140 DATA PCR,253,7,,PC,253,8,D,
--,243,1,X,-,242,1,Y,X,159,,U,Y
,191,,S,U,223,,PC
150 DATA S,255,,,++,241,1,,+,24
0,1,A,A,246,1,B,B,245,1,CC,D,25
1,1,DP
160 FOR J=0 TO 9:READ PU$(J),PU
(J):NEXT
170 DATA PC,128,U,64,S,64,Y,32,
X,16,DP,8,D,6,B,4,A,2,CC,1
180 CLS:PRINT @43,"ASSEMBLER"R$
R$TAB(8)"G=LER DO GRAVADOR"R$$S
TAB(8)"S=SALVAR NO GRAVADOR"R$R
$TAB(8)"A=MONTAR"
190 PRINT @296,"E=EDITAR LINHA"
R$R$TAB(8)"D=APAGAR LINHA"R$R$T
AB(8)"L=LISTAR NA TELA"
200 A$=INKEY$:IF A$="" THEN 200
210 JJ=INSTR("GSEDLA",A$)
220 IF JJ=0 THEN PRINT "<"A$">I
NVALIDO":FOR J=1 TO 1000:NEXT:G
OTO 180
230 CLS:ON JJ GOSUB 1420,1450,1
490,1710,1760,280
240 PRINT @0,"PRESSIONE <ENTER>
 PARA CONTINUARQUALQUER OUTRA T
ECLA PARA MENU  PRINCIPAL"
250 A$=INKEY$:IF A$="" THEN 250
260 IF A$<>R$ THEN 180
270 ON JJ GOSUB 1420,1450,1700,
1710,1850,290:GOTO 240
280 K=0:K9=0:P0=0
290 PS=0
300 PS=PS+1:IF PS<4 THEN K=K0:P
=P0:PRINT @1,"INICIO DA ETAPA"P
S:GOTO 330
310 P0=P:RETURN
320 IF PS=3 THEN PRINT " ERRO D
E POSBYTE"
330 GOSUB 1320
340 GOSUB 1260:OP$=C$:IF LEFT$(
OP$,1)="*" AND PS=3 THEN PRINT
OP$
350 IF LEFT$(OP$,1)="*" THEN 33
0
360 IF OP$="END" AND PS=3 THEN
PRINT:PRINT"    FIM.ULTIMO ENDE
RECO";P-1
370 IF OP$="END" THEN 300
380 IF OP$<>"ORG" THEN 420
390 GOSUB1260:S=0:IF LEFT$(C$,1
)="*" THEN S=P:C$=MID$(C$,2)
400 P=VAL(C$)+S:IF PS=3 THEN PR
INT:PRINT"     ORG";P
410 GOTO 330
420 IF P=0 AND PS=3 THEN PRINT"
 FALTA ORG":P=35000
430 CC=0:DF=0
440 C=INSTR(SK$(CC),OP$):IF C<>
0 THEN GOSUB 530:GOTO 570
450 C=INSTR(SK$(CC),LEFT$(OP$,3
)):IF C<>0 AND RIGHT$(OP$,1)<"C
" AND MID$(SK$(CC),C+3,1)<"A" T
HEN GOSUB 530:GOTO 540
460 IF C<>0 AND RIGHT$(OP$,1)="
B" GOSUB 530:GOTO 560
470 C=INSTR(SK$(CC),RIGHT$(OP$,
3)):IF C<>0 AND LEFT$(OP$,1)="L
" GOSUB 530:GOTO 550
480 CC=CC+1:IF CC<2 THEN 440
490 IF PS=3 THEN PRINT OP$
500 GOSUB 1350:Q3=Q2:RR(Q2)=P:I
F C$="" THEN 340
510 IF PS=3 THEN PRINT " LINHA
NAO RECONHECIDA"
520 GOTO 330
530 C=VAL(MID$(SK$(CC),C-2,2)):
RETURN
540 OP=K1(C)-32+16*(RIGHT$(OP$,
1)="A"):K2=0:GOTO 580
550 OP=K1(C)+4096:K2=5:GOTO 580
560 OP=K1(C)+64:K2=1:GOTO 580
570 OP=K1(C):K2=K2(C)
580 IF PS=3 THEN BY=P/256:GOSUB
 1240:BY=P-32768:GOSUB 1240:PRI
NT" "OP$;
590 IF OP>-1 THEN 740
600 GOSUB 1260:AD$=C$:IF PS=3 T
HEN PRINT " " LEFT$(AD$,10;
610 ON -OP GOTO 620,630,660,730
620 GOSUB 1860:IF NU=0 THEN RR(
Q3)=R:GOTO 330 ELSE 1050
630 IF PS=3 THEN PRINT TAB(20);
640 GOSUB 690:BY=255ANDR:GOSUB
1230
650 IF B$=AD$ THEN 330 ELSE 640
660 IF PS=3 THEN PRINT TAB(20);
670 GOSUB 690:BY=INT(R/256):GOS
UB 1230:BY=R-256*BY:GOSUB 1230
680 IF B$=AD$ THEN 330 ELSE 670
690 NN=INSTR(AD$,","):IF NN=0 T
```

```
HEN B$=AD$:GOTO 710
700 B$=LEFT$(AD$,NN-1):AD$=MID$
(AD$,NN+1)
710 IF LEFT$(B$,1)="$" THEN R=V
AL("&H"+MID$(B$,2))ELSE R=VAL(B
$)
720 RETURN
730 GOSUB 1860:IF NU=0 THEN P=P
+R:GOTO 330 ELSE 1050
740 B=239:IF K2=0 THEN 1140
750 GOSUB 1260:AD$=C$:IF PS=3 T
HEN PRINT " " LEFT$(AD$,9);
760 IF K2>3 THEN 1040
770 IF(K2<>3 OR (LEFT$(OP$,2)="
LD")) AND LEFT$(AD$,1)="#" THEN
 AD$=MID$(AD$,2):DF=1:OP=OP-48:
GOTO 1040
780 IF RIGHT$(AD$,1)="]" THEN A
D$=MID$(AD$,2,LEN(AD$)-2):B=B+1
6
790 IF OP<52 OR OP>55 THEN 870
800 K2=1:R=0:AA$=AD$
810 NN=INSTR(AA$,","):IF NN=0 T
HEN U$=AA$:GOTO 830
820 U$=LEFT$(AA$,NN-1):AA$=MID$
(AA$,NN+1)
830 IF U$=RIGHT$(OP$,1)THEN 320
840 NN=-1:FOR J=0 TO 9:IF U$=PU
$(J) THEN NN=J
850 NEXT:IF NN<0 THEN 320
860 R=R ORPU(NN):IF U$=AA$ THEN
 1140 ELSE 810
870 K2=3:C=INSTR(AD$,","):IF C=
0 THEN 1040
880 IF OP<>30 AND OP<>31 THEN 9
50
890 NN=INSTR(AD$,","):U$=LEFT$(
AD$,NN-1):GOSUB 930:N1=H-1:IF H
=0 THEN 320
900 U$=MID$(AD$,NN+1):GOSUB 930
:N2=H-1:IF H=0 THEN 320
910 IF(8ANDN1)<>(8ANDN2) THEN 3
20
920 R=16*N1+N2:K2=1:GOTO 1140
930 FOR J=1 TO 12:IF Y$(J)=U$ T
HEN H=J
940 NEXT:RETURN
950 C2=C+1:FOR J=0 TO 12:L=LEN(
X$(J)):IF MID$(AD$,C2,L)<>X$(J)
 THEN 980
960 IF (B ORV(J)) AND 239<239 T
HEN 320
970 C2=C2+L:B=B AND V(J):IF KK(
J) THEN K2=KK(J)-1
980 IF J=9 THEN J2=C2:C2=C2+(C2
-1)*(K2<6)
990 NEXT
1000 IF(15 AND B)=15 THEN B=B-6
1010 IF PS=3 AND J2<=LEN(AD$)AN
D K2<>7 THEN PRINT "ERRO DE IND
EXACAO":GOTO 330
1020 IF(K2=0 AND C>2) OR (MID$(
AD$+",",J2,1)<>",") AND PS=3 TH
EN PRINT "ERRO DE ENDERECAMENTO
":GOTO 330
1030 AD$=LEFT$(AD$,C-1):IF J2=C
 THEN R=0:GOTO 1060
1040 GOSUB 1860:IF NU=0 THEN 10
60
1050 IF PS=3 THEN PRINT" ENDERE
CAMENTO NAO ENTENDIDO"
1060 IF K2=7 THEN K2=3:GOTO 108
0
1070 IF K2>3 THEN R=R-P-2+(OP>2
55)+(B<>239)+(K2>4):R=R-((K2=6)
AND(R>-129)AND(R<128)):K2=K2-3
1080 IF B=239 AND K2=3 THEN K2=
2:IF R<256 AND DF=0 THEN K2=1:O
P=OP-32:IF(240 AND OP)=80 THEN
OP=OP-80
1090 IF PS=3 AND((OP>31 AND OP<
48)OR OP=141)AND(R<-128 OR R>12
7)THEN PRINT " DESVIO FORA DE F
AIXA";:GOTO 330
1100 IF B=255 THEN B=159:K2=2
1110 IF B<>239 THEN OP=OP-16:IF
 K2=3 THEN K2=2:IF ABS(R+.5)<12
```

```
8 THEN K2=1:B=B-1:R=255 AND R
1120 IF(15 AND B)=8 AND R=0 THE
N B=B-4:K2=0
1130 IF(31 AND B)=8 AND (R<16 O
R R>239) THEN B=(B-136)OR(31 AN
D R):K2=0
1140 IF PS=3 THEN PRINT TAB(20)
;
```

```
1150 IF OP=>0 THEN BY=OP/256:GO
SUB 1220:BY=OP:GOSUB 1230
1160 IF B<>239 THEN BY=B:GOSUB
1230
1170 IF K2=0 THEN 330
1180 GOSUB 1200:IF K2=2 THEN BY
=R/256:GOSUB 1230
1190 BY=R-256*INT(R/256):GOSUB
1230:GOTO 330
1200 IF PS=3 THEN PRINT " ";
1210 RETURN
1220 IF INT(BY)=0 THEN RETURN
1230 P=P+1:IF PS=3 THEN POKE P-
1,255 AND BY
1240 BY=255 AND BY:IF PS=3 THEN
 PRINT RIGHT$("0"+HEX$(BY),2);
1250 RETURN
1260 IF K>N THEN C$="END":RETUR
N
1270 K1=K9+1:IF K9>=LEN(T$(K))
THEN C$=" FALTA MNEMONICO":RETU
RN
1280 K9=K1:IF MID$(T$(K),K1,1)=
" " THEN 1270
1290 IF K9>LEN(T$(K))THEN C$=MI
D$(T$(K),K1,K9-K1):RETURN
1300 IF MID$(T$(K),K9,1)<>" " T
HEN K9=K9+1:GOTO 1290
1310 C$=MID$(T$(K),K1,K9-K1):RE
TURN
1320 IF K9<=LEN(T$(K))AND PS=3
THEN PRINT RIGHT$(T$(K),LEN(T$(
K))-K9+1);
1330 K=K+1:K9=0:IF PS=3 THEN PR
INT
1340 RETURN
1350 X$=""
1360 IF C$<"A" OR C$>="[" THEN
1380
1370 X$=X$+LEFT$(C$,1):C$=MID$(
C$,2):GOTO 1360
1380 IF C$<>"" THEN RETURN
1390 FOR Q2=1 TO VV:IF X$=Z$(Q2
) THEN 1410
1400 NEXT:VV=VV+1:Z$(VV)=X$:Q2=
VV:RR(VV)=23000
1410 RETURN
1420 CLS:MOTORON:PRINT @161,"PO
SICIONE O GRAVADOR, PRESSIONE Q
UALQUER TECLA E APERTE <PLAY>."
1430 U$=INKEY$:IF U$="" THEN 14
30
1440 OPEN"I",#-1,"ASM":PRINT" C
ARREGANDO PROGRAMA":INPUT #-1,N
:FOR J=1 TO N:INPUT #-1,T$(J):N
EXT:CLOSE #-1:RETURN
1450 CLS:MOTORON:PRINT @161,"PO
SICIONE O GRAVADOR , APERTE   <
RECORD> E PRESSIONE QUALQUER
TECLA."
1460 U$=INKEY$:IF U$="" THEN 14
60
1470 OPEN"O",#-1,"ASM":PRINT" G
RAVANDO PROGRAMA":PRINT #-1,N:F
OR J=1 TO N:PRINT #-1,T$(J):NEX
T:CLOSE #-1:RETURN
1480 PRINT #-1,N:FOR J=1 TO N:P
RINT#-1,T$(J):NEXT:CLOSE#-1:RET
URN
1490 PRINT " INTRODUZA O NUMERO
 DA LINHA (LI NHAS NUMERADAS DE
 DEZ EM DEZ)"
1500 INPUT K:CLS
1510 K2=K/10:IF K2>N THEN K2=N+
1:N=N+1:T$(K2)="":PRINT @480,""
1520 IF K2<.1 THEN K2=.1
1530 IF K2=INT(K2) THEN 1550
1540 K2=INT(K2)+1:FOR K3=N TO K
2-1 STEP -1:T$(K3+1)=T$(K3):NEX
T:N=N+1:T$(K2)=""
1550 P1=1478:P0=P1
1560 PRINT @448,K;TAB(6)T$(K2):
P9=P0+LEN(T$(K2))
1570 IF P1<P0 THEN P1=P0
1580 IF P1>P9 THEN P1=P1-1
1590 P8=PEEK(P1):POKE P1,63 AND
 P8
1600 P7=0:A$=INKEY$:IF A$="" TH
EN 1600
1610 IF A$=R$ THEN POKE P1,P8:R
ETURN
1620 IF A$=CHR$(9) THEN POKE P1
,P8:P1=P1+1:GOTO 1580
1630 IF A$=CHR$(8) THEN POKE P1
,P8:P1=P1-1:GOTO 1570
1640 IF A$=CHR$(10) THEN A$="":
GOTO 1670
1650 IF A$=CHR$(94) THEN A$=" "
+MID$(T$(K2),P1-P0+1,1):P7=-1:G
OTO 1670
1660 IF A$<" " THEN 1600
1670 IF P1-P0+1>LEN(T$(K2)) THE
N T$(K2)=LEFT$(T$(K2),P1-P0)+A$
:GOTO 1690
1680 T$(K2)=LEFT$(T$(K2),P1-P0)
+A$+RIGHT$(T$(K2),LEN(T$(K2))-P
1+P0-1)
1690 P1=P1-(LEN(A$)>0)+P7:GOTO
1560
1700 PRINT @32,"":K=K+10:GOTO 1
510
1710 IF N=0 THEN CLS:PRINT" NAD
A A APAGAR":FOR C=1 TO 1000:NEX
T:RETURN
1720 CLS:PRINT"INTRODUZA NO DA
LINHA            (LINHAS NUMERAD
AS DE DEZ EM DEZ)"
1730 INPUT K:K2=K/10
1740 IF K2>N OR K2<1 OR K2<>INT
(K2) THEN PRINT" ESTA LINHA NAO
 EXISTE":RETURN
1750 K=K2:FOR K3=K2 TO N:T$(K3)
=T$(K3+1):NEXT:N=N-1:PRINT @95,
K*10;"  ";T$(K):RETURN
1760 IF N=0 THEN PRINT " NADA A
 LISTAR":FOR C=1 TO 1000:NEXT:R
ETURN
1770 PRINT" INTRODUZA  O NO DA
PRIMEIRA E  DA ULTIMA LINHA  (L
INHAS NUMERA-DAS EM   MULTIPLOS
 DE 10)"
1780 INPUT K,K2:K=INT(K):K2=INT
(K2):K1=K/10:K2=K2/10
1790 IF K2>N THEN K2=N
1800 IF K1<1 THEN K1=1
1810 IF K2<K1 AND K2=N THEN RET
URN
1820 IF K2<K1 THEN CLS:PRINT "
CONJUNTO DE LINHAS INVALIDO":GO
TO 1770
1830 CLS:PRINT @96,;:FOR K3=K1
TO K2:PRINT K3*10" "T$(K3):NEXT
1840 RETURN
1850 K=K2-K1:K1=K2+1:K2=K1+K:IF
 N=0 THEN 1760 ELSE 1790
```

```
1860 NU=0:R=0
1870 S=1
1880 IF AD$="" THEN RETURN
1890 X$=LEFT$(AD$,1):BD$=MID$(A
D$,2):IF X$="*" THEN R=R+P*S:AD
$=BD$:GOTO 1870
1900 IF X$="+" THEN AD$=BD$:GOT
O 1880
1910 IF X$="-" THEN AD$=BD$:S=-
S:GOTO 1880
1920 Q=0:IF X$<>"&" THEN 1950
1930 IF BD$>="0" AND BD$<"2" TH
EN Q=Q*2+ASC(BD$)-48:BD$=MID$(B
D$,2):GOTO 1930
1940 R=R+Q*S:AD$=BD$:GOTO 1870
1950 IF X$<>"$" OR BD$<"0" OR B
D$>"F" THEN 1980
1960 Q2=VAL("&H"+LEFT$(BD$,1)):
BD$=MID$(BD$,2):Q=Q*16+Q2:X$=LE
FT$(BD$,1)
1970 IF(X$>"/" AND X$<":") OR (
X$>"@" AND X$<"G") THEN 1960 EL
SE R=R+Q*S:AD$=BD$:GOTO 1870
1980 IF X$<"A" OR X$>"Z" THEN 2
010
1990 C$=AD$:GOSUB 1350:IF C$<>"
" GOSUB 1390
2000 R=R+RR(Q2)*S:AD$=C$:GOTO 1
870
2010 IF X$<"0" OR X$>"9" THEN R
=0:NU=1:RETURN
2020 IF AD$>"/" AND AD$<":" THE
N Q=Q*10+ASC(AD$)-48:AD$=MID$(A
D$,2):GOTO 2020
2030 R=R+S*Q:GOTO 1870
```

COMO FUNCIONA O PROGRAMA

Não se esqueça de reservar um espaço suficiente de memória para seu programa, usando **CLEAR** antes de rodá-lo.

Para introduzir o seu programa pressione a tecla E. Ele pedirá então um número de linha. Neste programa cada instrução em mnemônico deve ser introduzida em uma linha de BASIC. É preciso também que os números de linha sejam múltiplos de 10 e cada linha contenha apenas um comando Assembly com seus respectivos operandos.

A primeira linha deve abrigar o endereço inicial da porção da memória onde será colocada a rotina em código. É necessário que esse endereço esteja na área reservada por **CLEAR**. Para especificar sua origem, usamos o comando **ORG** seguido do endereço inicial.

Se um endereço inicial não for especificado, o programa tentará posicionar o Assembler a partir do endereço 35000, que pertence à memória ROM. O resultado não será satisfatório, uma vez que não se pode colocar códigos na ROM. O programa procede assim para evitar que o Assembler seja colocado em uma área prejudicial ao funcionamento da máquina. Quando o endereço acima é usado, o programa comunica: "origem não encontrada".

Normalmente, são utilizados os mnemônicos padrão do 6809. Números hexadecimais devem ser precedidos de $ (cifrão), números binários, de % (porcento). Algarismos sem prefixo são considerados decimais.

Alguns Assembler para o TRS-Color reconhecem códigos ASCII, se estes forem precedidos de ' (apóstrofo) ou ! (ponto de exclamação). Não é o caso de nosso programa. Assim, o comando **FCC**, que manipula caracteres ASCII, não pode ser usado.

Para editar linhas, empregue as teclas do cursor. As setas "direita" e "esquerda" movimentam o cursor ao longo da linha; a seta "para cima" insere e a seta "para baixo" apaga.

A última linha do programa deve ser **END**; o Assembler criará uma linha desse tipo, caso esqueçamos de fazê-lo.

Se, depois de digitarmos uma linha, pressionarmos outra tecla que não **ENTER**, o programa voltará ao menu. Se então pressionarmos L, os mnemônicos serão listados para uma conferência.

Caso haja algum erro, retorne ao menu e pressione E. Especifique então o número da linha a ser corrigida.

Se quisermos inserir uma nova linha entre duas já existentes, devemos entrar no modo de edição e dar a ela um número intermediário. Uma nova listagem mostrará a nova linha no lugar correto, sendo que todas as linhas terão números múltiplos de 10. Apenas uma linha de cada vez pode ser inserida dessa maneira.

Para apagar uma linha, retorne ao menu e pressione D. Especifique então o número da linha a ser apagada. Quando não houver mais modificações a fazer, retorne ao menu e pressione A. O programa em Assembly será então "montado" na memória. Quando o processo terminar, o endereço final do programa em código será mostrado na tela.

A opção de gravação — tecla S — grava apenas o programa em Assembly ou *programa fonte*. Para gravar o Assembler, use o caminho normal. Para gravar o programa em código — ou *programa objeto* — é necessário sair do Assembler usando a tela **BREAK**:

```
CSAVEM "NOME", INICIO, FIM,
DEFEXEC
```

NOME é a denominação do programa; **INÍCIO** é o endereço inicial definido por **ORG**; e **FIM**, o endereço final fornecido pelo programa após a montagem.

O último número, **DEFEXEC**, diz ao TRS-Color por onde começar a executar a rotina em código. Geralmente, ele é igual ao endereço inicial. Agora podemos montar qualquer programa a partir de sua listagem em Assembly.

MICRO DICAS

COMO ENCONTRAR ERROS EM PROGRAMAS LONGOS

Mesmo o programador mais experiente terá problemas em digitar programas longos como esse Assembler. Não importa a destreza de seus dedos: fatalmente, em algum lugar, um erro será cometido.

Muitos desses erros são encontrados quando se confere a listagem. As mensagens de erro do micro ajudarão também, se procurarmos saber o que significam.

Tais mensagens, contudo, podem não ser suficientes para detectar o local de um erro em um programa muito longo. Nesses programas, uma linha pode ser executada sem problemas muitas vezes, só vindo a provocar um erro quando uma variável assumir um valor incompatível devido a um erro de digitação cometido em outra parte do programa.

Felizmente, o TRS-Color conta com uma função de rastreamento — *trace* —, que facilita a localização de erros.

Tal função é ativada pelo comando **TRON**, que deve ser usado no modo imediato, sem número de linha. Quando rodamos o programa, ela nos mostra o número da linha que está sendo executada.

Para desativar a função de rastreamento, use o comando **TROFF**.

UM TESTE

Para testar o Assembler, digite o programa de deslocamento da tela para a direita apresentado na página 219. Esse programa resultará nos seguintes códigos:

```
8E 06 00 E6 82 34 04 C6 1F A6
82 A7 01 5A 26 F9 35 04 E7 84
8C 04 00 2E EA 39
```

Em algumas das máquinas compatíveis com o TRS-Color é possível aumentar a velocidade do programa acrescentando **POKE 65495,0** no início da linha 180. Se isto for feito, devemos modificar outras linhas a fim de que a alta velocidade não prejudique a gravação em fita. Assim, adicione **POKE 65494,0** ao início das linhas 1420 e 1450.

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

PROGRAMAÇÃO BASIC

Os computadores são capazes de executar milhões de operações lógicas por segundo. Veja como isso funciona.

PROGRAMAÇÃO DE JOGOS

Incorpore objetos ao programa de aventura e enriqueça seu vocabulário com novos verbos.

PROGRAMAÇÃO BASIC

Aprenda a localizar e a corrigir os erros de um programa à medida que eles aparecem e evite dores de cabeça.

CÓDIGO DE MÁQUINA

Figuras móveis produzidas no Apple e no ZX-81. Como movimentar os desenhos.