COLOR BASIC

COLOR BASIC

COLOR BASIC

INSTRUÇÃO PROGRAMADA

SEJAM BEM VINDOS — NOVATOS !!!

Se você não sabe nada sobre computadores e gostaria de não perder muito tempo com explicações técnicas, relaxe, este manual é para você.

Usando-o como guia, você será capaz de conhecer este moderno invento da tecnologia e apreciar imediatamente o seu computador.

Você sentirá, especialmente no início, que estamos fazendo você trabalhar muito com jogos, canções e outros tipos de programas para divertimento, mas não se preocupe. Se você fizer os exercícios práticos, terá muito com que se divertir. Começamos com a parte divertida dos programas por ser este o meio mais rápido de aprender a trabalhar com o computador. Quando sentir que ele complementa sua vida, você será capaz de fazê-lo trabalhar em qualquer coisa que desejar.

Então, sente-se e perca algumas horas com ele.

Digite o que quiser, brinque com ele, tente fazê-lo executar as coisas mais estranhas, pois há um número ilimitado de coisas que ele pode fazer para você.

... E ALÔ — VELHOS AMIGOS !!!

Não nos esquecemos de você; se já sabe programar, vá direto ao apêndice G. Lá você encontrará um sumário do COLOR BASIC com o número da página que deve consultar para conhecer, em detalhes, cada instrução.

PARA DAR A PARTIDA

Ligue o computador conforme orientação fornecida no manual de operação:

- Ligue a televisão

- Selecione o canal 3 ou 4

- Ligue o computador. O botão on/off (liga/desliga) fica na parte detrás, à esquerda de quem olha para ele.

Uma vez ligado, aparecerá esta mensagem:

```
COLOR BASIC 1.1
C O L O R  6 4
OK
```

Se esta mensagem não aparecer, ajuste o brilho e o contraste da televisão. Se ainda assim nada acontecer, desligue o equipamento, verifique as conexões e torne a ligá-lo. Para maiores esclarecimentos, consulte o capítulo referente a defeitos e reparos no manual de operação.

Uma vez obtida esta mensagem, você pode iniciar.

COMO CONVERSAR COM O COMPUTADOR?

Neste manual você aprenderá a conversar com o computador, isto é, a programá-lo. Sabendo comunicar-se com ele, você será capaz de ordenar-lhe o que quiser. Não se preocupe, vai ser fácil.

O computador entende uma linguagem chamada COLOR BASIC que é uma versão do BASIC (Beginners All-purpose Symbolic Instruction Code). Existem muitas linguagens de computador, mas COLOR BASIC é a linguagem que o seu computador entende.

Apresentaremos as palavras BASIC numa forma fácil de aprender. Quando atingir metade do manual, você poderá ter esquecido o significado de algumas palavras e caso isto ocorra, simplesmente consulte o apêndice G, neste mesmo manual.

CONTEÚDO

SEÇÃO III- APLICAÇÕES COMERCIAIS

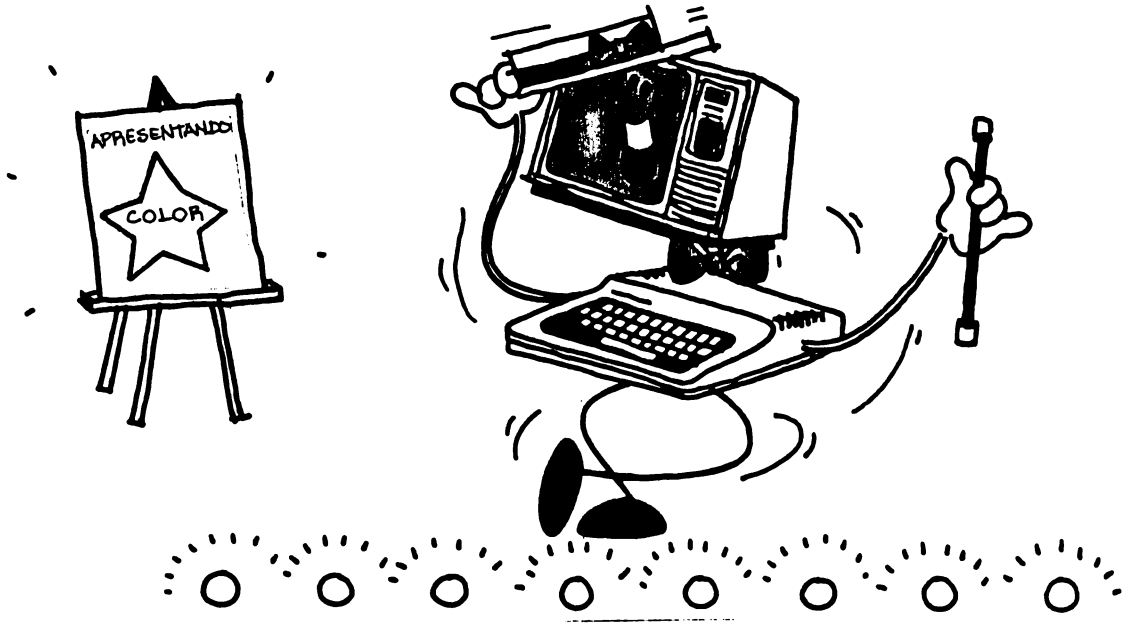
APÊNDICES

PREPARE — SE

PARA

RECEBÊ — LO

CAPÍTULO I



CONHEÇA SEU COMPUTADOR

1

CONHEÇA SEU COMPUTADOR

O seu computador está conectado? Ligado? Pronto para o primeiro trabalho?

Nestes dois primeiros capítulos, vamos apresentar o computador, o modo como pensa, alguns dos seus talentos e até mesmo um pouco da sua sutileza. Quando terminá-los, você estará pronto para programar.

Digite à vontade no teclado e então pressione a tecla (ENTER).

Preocupe-se apenas com a última linha da tela. Ela estará assim:

```
OK
```

OK, o computador deu a dica, ele está dizendo "OK, chega de loucuras...". Ele, calmamente, espera pelo seu comando. Você agora é o mestre e poderá dizer-lhe para fazer o que desejar.

Todas as letras digitadas devem aparecer, na tela, em cor preta com fundo verde, se acontecer o contrário, (isto é, fundo preto e letras verdes), pressione as teclas (SHIFT) e (0) (zero), simultaneamente.

Está vendo a luz piscando na tela? Você só pode digitar alguma coisa quando vir isto.

OLA, SOU O SEU COMPUTADOR

Forneça o primeiro comando. Digite o seguinte:

```
PRINT "OLA,SOU O SEU COMPUTADOR"
```

Agora verifique a linha. Colocou as aspas no lugar correto? Caso tenha cometido um erro, não se preocupe, pressione a tecla (←) e o último caráter digitado desaparecerá. Pressione-a outra vez e o caráter anterior também desaparecerá, e assim por diante.

Pronto? Deve aparecer na tela:

```
OK
PRINT "OLA,SOU O SEU COMPUTADOR"
```

Pressione (ENTER) e observe. Aparecerá na tela:

```
OK
PRINT "OLA,SOU O SEU COMPUTADOR"
OLA,SOU O SEU COMPUTADOR
OK
```

O computador obedeceu a ordem e imprimiu a mensagem colocada entre aspas.

Forneça outra mensagem para ele imprimir. Digite:

```
PRINT "2"
```

Pressione (ENTER). Mais uma vez o computador obedece e imprime:

2

Tente outra ordem:

PRINT "2 + 2" (ENTER)

Ele obedece e imprime:

2 + 2

Você provavelmente espera mais do computador, como, por exemplo, uma resposta. Bem, tente sem aspas.

Digite:

PRINT 2 + 2 (ENTER)

Muito melhor. Agora o computador imprimiu a resposta:

4

Logicamente, as aspas devem ter um significado. Faça outras experiências, digite cada uma destas linhas:

O computador interpreta as aspas como um jornalista o faz. Se a mensagem não estiver entre aspas, ele a interpreta como uma operação aritmética entre números.

```
PRINT 5 + 4 (ENTER)
PRINT "5 + 4" (ENTER)
PRINT "5 + 4 E' IGUAL A" 5 + 4 (ENTER)
PRINT 6/2 "E' 6/2" (ENTER)
PRINT "8/2" (ENTER)
PRINT 8/2 (ENTER)
```

Chegou a alguma conclusão a respeito das aspas?

DIFERENÇAS ENTRE STRINGS E NÚMEROS

O computador interpreta tudo o que você digita como string ou número. Se a mensagem estiver entre aspas é um string e o computador a vê exatamente como é. Se não estiver entre aspas, é um número e o computador a interpreta como expressão numérica.

UMA CALCULADORA COLORIDA

O computador resolve facilmente qualquer operação aritmética. Deixe-o fazer uma divisão complicada.

```
PRINT "3862 DIVIDIDO POR 13.2 E' " 3862/13.2  (ENTER)
```

Vamos fazer um exercício de multiplicação:

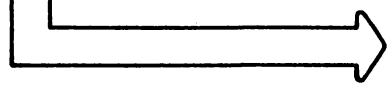
```
PRINT 1589 *2 (ENTER)
```

Observe que o sinal de multiplicação, para o computador, é um asterisco (*) e não o sinal X normalmente usado em matemática.

Verifique como o computador manipula os dados entre aspas e os que não estão entre aspas.

Isto porque o computador é uma criatura tão precisa que confundiria o sinal de multiplicação (X) com a letra X do alfabeto.

Tente um pouco mais:



```
PRINT "15 *  2 = " 15 * 2 (ENTER)
PRINT 18 *  18 " E' O QUADRADO DE 18" (ENTER)
PRINT 33.3/22.82 (ENTER)
```

Agora é a sua vez. Escreva dois comando que imprimam estas duas operações matemáticas:

157 / 13.2 =
95 * 43 =

FAÇA VOCÊ MESMO OS COMANDOS

Se usou corretamente os comandos, o computador imprimiu o seguinte:

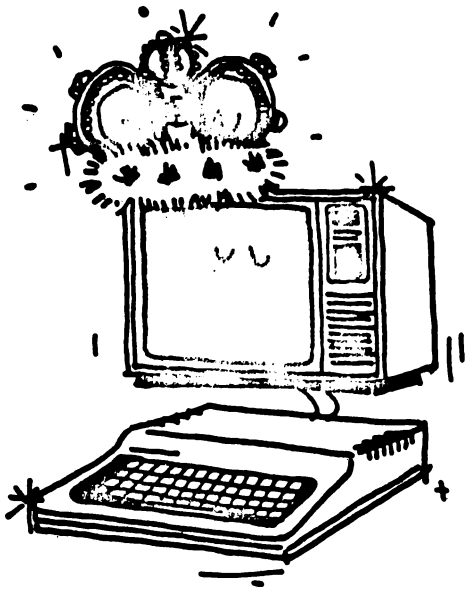
*Não se esqueça que a multiplicação é representada por um * e não por um X .*

157/13.2 = 11.8939394
95 * 43 = 4085

Pronto para as respostas? Veja:

```
PRINT "157 / 13.3 = " 157/13.2
PRINT "95 * 43 = " 95 * 43
```

ELE TEM SUAS REGRAS...



A esta altura o computador, provavelmente, já imprimiu algumas mensagens engraçadas na tela. Se não, digite, deliberadamente, a palavra PRINT, de maneira errada:

PRIINT "OLA" (ENTER)

O computador responde:

?SN ERROR

SN ERROR significa erro de sintaxe. É a forma do computador dizer que o comando PRIINT não existe em seu vocabulário... "Eu não tenho a mínima idéia do que você quer dizer". Sempre que ocorrer um erro de sintaxe, provavelmente, você, cometeu um erro de digitação.

O computador também imprime uma mensagem de erro quando apesar de entender o que você disse, ele considera o seu pedido ilógico ou impossível de executar. Por exemplo, tente isto:

PRINT 5/0 (ENTER)

O computador imprime:

?/0 ERROR

Que significa: "não me peça para dividir por zero isto é impossível!"

Caso ocorra uma mensagem de erro que você não entenda, consulte o apêndice F. Lá, estão relacionadas todas as mensagens de erro e suas prováveis causas.

ELE GOSTA DE APARECER

Até aqui, tudo que o computador fez, foram impressões silenciosas numa tela verde, mas ele gosta de aparecer. Digite:

CLS (3) (ENTER)

Se você não obtiver a cor exata, consulte o manual de operação.

Agora a tela apresenta uma bonita cor azul com uma lista verde no alto. O comando digitado disse ao computador para limpar a tela e torná-la de cor nº 3 (azul).

Pressione (ENTER) para ver na tela a mensagem OK.

Digite:

CLS (7) (ENTER)

Agora você deve ter uma tela carmim com uma lista verde no alto. Tente outras cores, use qualquer número de 0 a 8. O computador tem 9 cores e cada uma tem um código.

ERRO - Se ocorrer uma mensagem ?FC ERROR é porque foi usado um número fora do intervalo 0 a 8.

Digite CLS sem código de cor:

CLS (ENTER)

Se fizer isto, o computador entende que você, simplesmente, deseja limpar a tela e manter o fundo verde.

EIS O SOM DO COMPUTADOR _ 1, 2...

Digite o seguinte:

SOUND 1, 100 (ENTER)

Cuidado para não aumentar demais o volume da televisão.

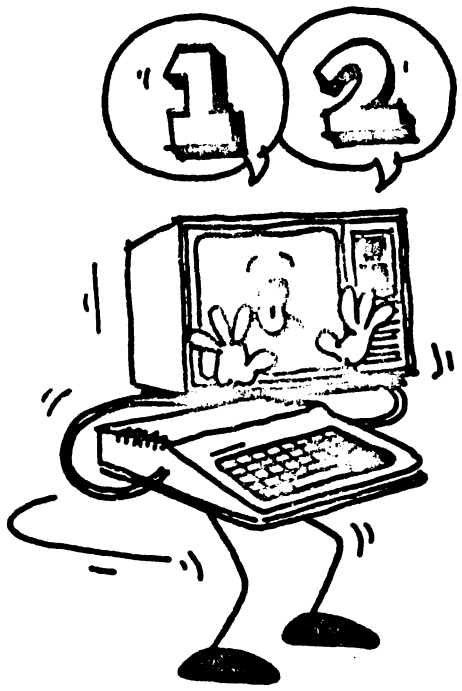
Se você não ouvir nada, aumente o volume da televisão e tente outra vez.

O que você está ouvindo são seis segundos do tom mais baixo que o computador pode emitir.

Para ouvir o mais alto, digite:

SOUND 255, 100 (ENTER)

Muito bem, ele possui uma faixa de tons... Tente outros números. Esperamos que goste da voz do computador.



Quer saber qual o significado do outro número? (Talvez já tenha descoberto!). O segundo número diz ao computador por quanto tempo deve durar o som. Você pode usar números de 1 a 255. Tente 1:

SOUND 128, 1 (ENTER)

O computador emitirá um tom com duração de seis centésimos de segundo. Tente 10:

SOUND 128, 10 (ENTER)

O computador emitirá um tom de seis décimos de segundo.

Tente variações dos dois números, mas mantenha-os sempre entre 1 e 255.

ERRO - Se ocorrer a mensagem ?FC ERROR, é porque você usou um número fora do intervalo 1 a 255.

UMA COISA A MAIS ...

Não deixe de ler esta nota, ela é muito importante.

Pressione simultaneamente as teclas (SHIFT) e (0) (zero) e digite algumas letras. As letras digitadas aparecerão verdes em fundo preto.

Agora com as cores "invertidas", pressione (ENTER) e então digite:

PRINT "OLA" (ENTER)

O computador acusa ?SN ERROR. Ele não entendeu o comando.

Pressione (SHIFT) e (0), outra vez, e digite algumas letras. Tudo voltou ao normal, isto é, letras pretas em fundo verde.

Pressione (ENTER) e digite a linha novamente. Agora ele funcionou.

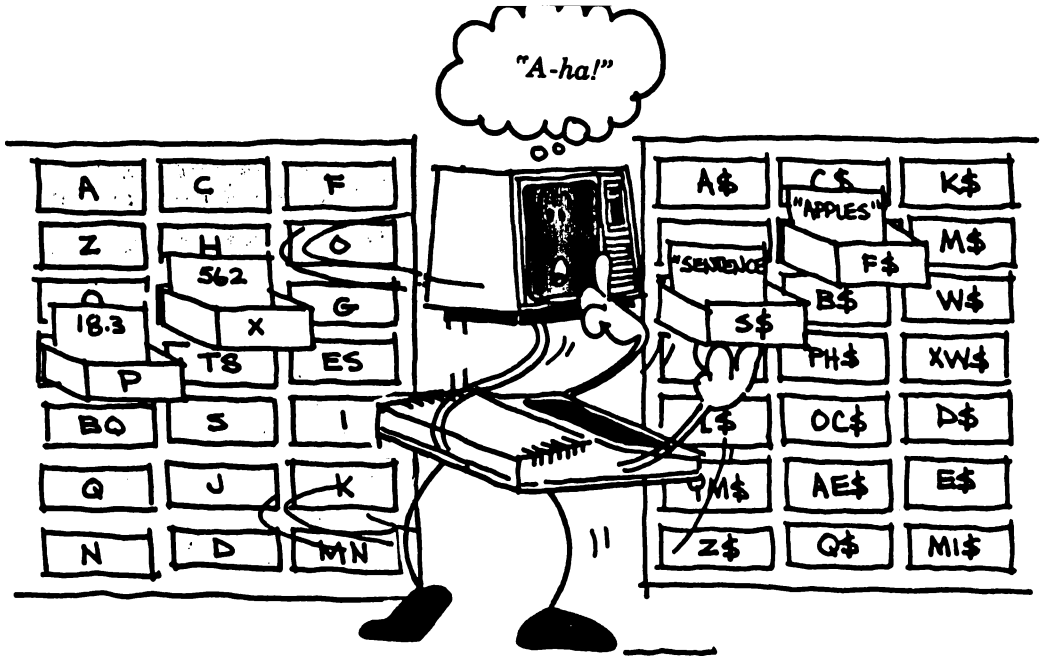
Este assunto foi abordado, para o caso de você pressionar (SHIFT) e (0) sem querer. O computador não entende nenhum comando digitado em cores invertidas. Se isto ocorrer, pressione (SHIFT) e (0) para voltar ao normal.

A S S U N T O S A P R E N D I D O S N O C A P Í T U L O 1

COMANDOS BASIC	TECLADO	CONCEITOS
PRINT		strings X números
SOUND	←	mensagens de erro
CLS	(ENTER)	

Colocaremos uma lista como esta ao final de cada capítulo para você não esquecer os comandos aprendidos.

CAPÍTULO 2



SEU COMPUTADOR NUNCA ESQUECE

(...A MENOS QUE VOCÊ O DESLIGUE...)

2

SEU COMPUTADOR NUNCA ESQUECE

(... A MENOS QUE VOCÊ O DESLIGUE...)

Uma das características que torna o computador poderoso é a sua capacidade de lembrar qualquer coisa que você lhe pergunte. Para fazê-lo lembrar o número 14, digite:

A = 14 (ENTER)

Agora digite qualquer coisa para confundí-lo. Quando tiver feito isto, pressione (ENTER). Para ver se ele não esqueceu o valor de A, digite:

PRINT A (ENTER)

O computador se lembrará do número 14 enquanto estiver ligado... ou até você fazer o que faremos agora. Digite:

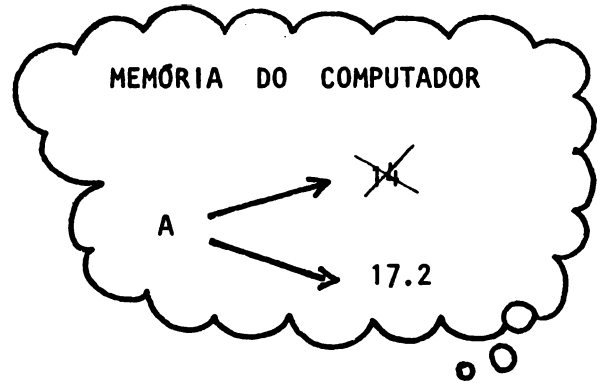
A = 17.2 (ENTER)

Se mandar imprimir A (PRINT A), ele imprimirá o número 17.2.

Ele ficou confuso ou se esqueceu?

Se você já conhece Basic, você pode estar acostumado a usar a palavra LET antes da linha de atribuição. Este computador acha a palavra LET desnecessária e fica confuso quando ela é usada.

Isto foi exatamente o que aconteceu na memória do computador:



Não é preciso usar a letra A. Pode-se usar qualquer outra de A a Z. (Na verdade, você pode usar duas letras quaisquer).

Tente isto:

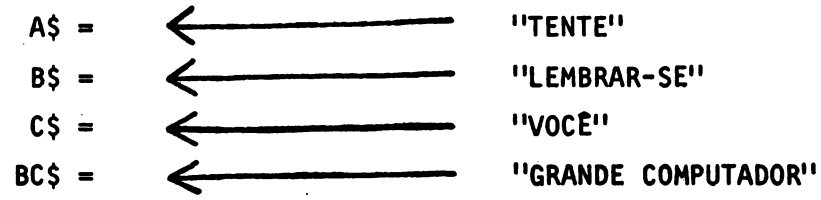
```
B = 15  (ENTER)
C = 20  (ENTER)
BC = 25 (ENTER)
```

Faça-o imprimir todos os números. Digite:

```
PRINT A, B, C, BC
```

Para ele se lembrar de um string de letras ou números, coloque o sinal cifrão (dólar) logo após a letra. Digite:

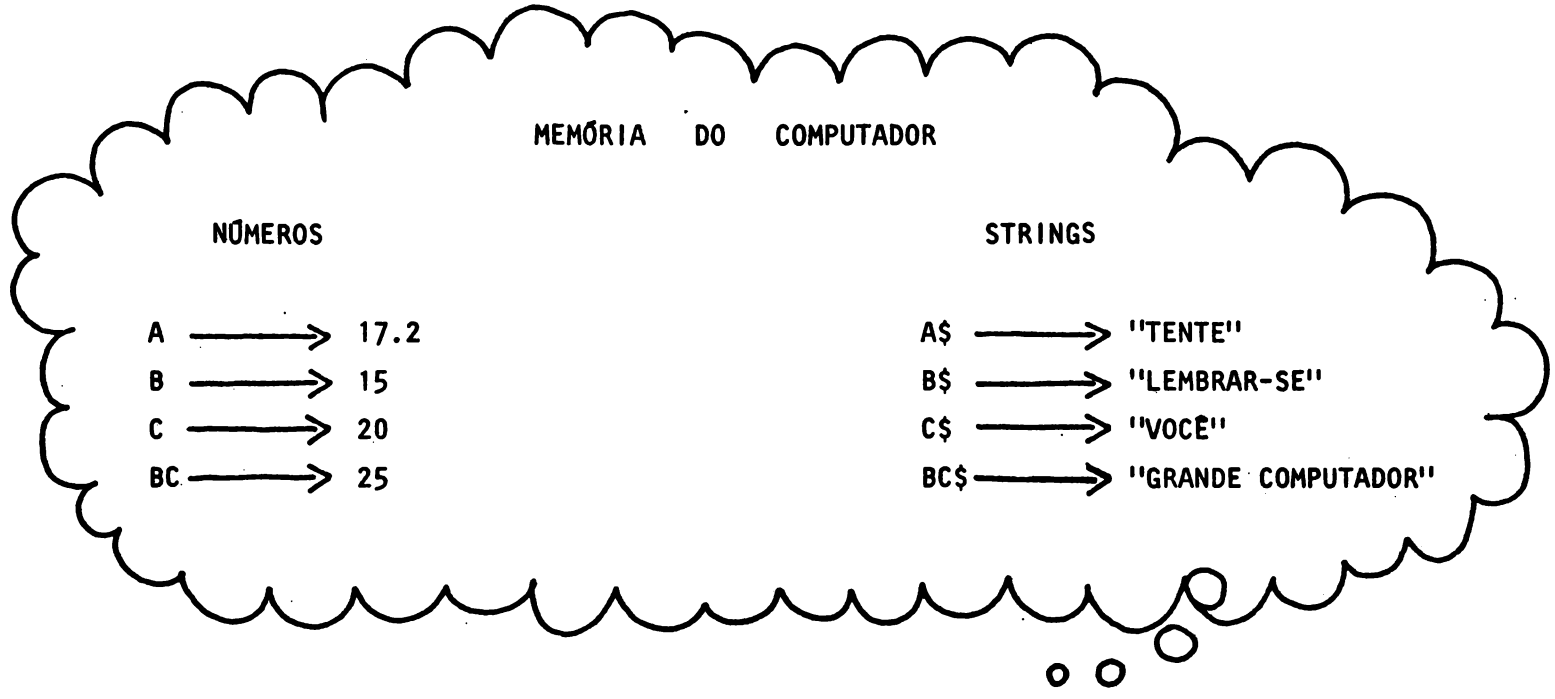
O sinal cifrão (dólar) significa, para o computador, que a variável é do tipo string.



Vamos ver como ele é inteligente. Digite:

PRINT A$, B$, C$, BC$ (ENTER)

O computador chama todas essas letras de variáveis. Até aqui nós usamos as seguintes variáveis:



Tente fazer o computador se lembrar de uma letra que você ainda não tenha digitado. O que ocorre... interessante...

Tente checar estas variáveis para ver se o computador se lembra dos seus conteúdos. Por exemplo, digite:

```
PRINT BC  (ENTER)
```

Para ver se BC ainda contém 25.

Imagine estas variáveis como caixas onde você consegue armazenar suas informações. Algumas das caixas são para strings e outras para números e você usa estas variáveis para dar nomes às caixas.

Como dissemos anteriormente, o computador tem suas próprias regras e pode confundir-se caso você não as obedeça.

Você pensa que o computador aceitará estas linhas?

```
D = "6"  (ENTER)
Z = "ISTO E' UM STRING"  (ENTER)
```

TM significa TYPE MISMATCH ERROR, ou seja, que você não seguiu as regras do computador.

Em ambos os casos ele responde ?TM ERROR. Ele está dizendo que você jogou de acordo com as suas regras, mas não com as dele.

Estas são as regras que você ignorou:

R E G R A S P A R A S T R I N G S

(1) Qualquer dado entre aspas é considerado string.
(2) Strings só podem ser designados por variáveis com um sinal $ após o nome.

É preciso obedecer as regras do computador

Para obedecer às regras do computador, temos que colocar um sinal $ (dólar) após as letras D e Z. Digite:

```
D$ = "6"  (ENTER)
Z$ = "ISTO E' UM STRING"  (ENTER)
```

Você pensa que o computador aceitará isto?

```
D$ = 6  (ENTER)
```

Estas são as regras que foram ignoradas:

R E G R A S P A R A N Ú M E R O S

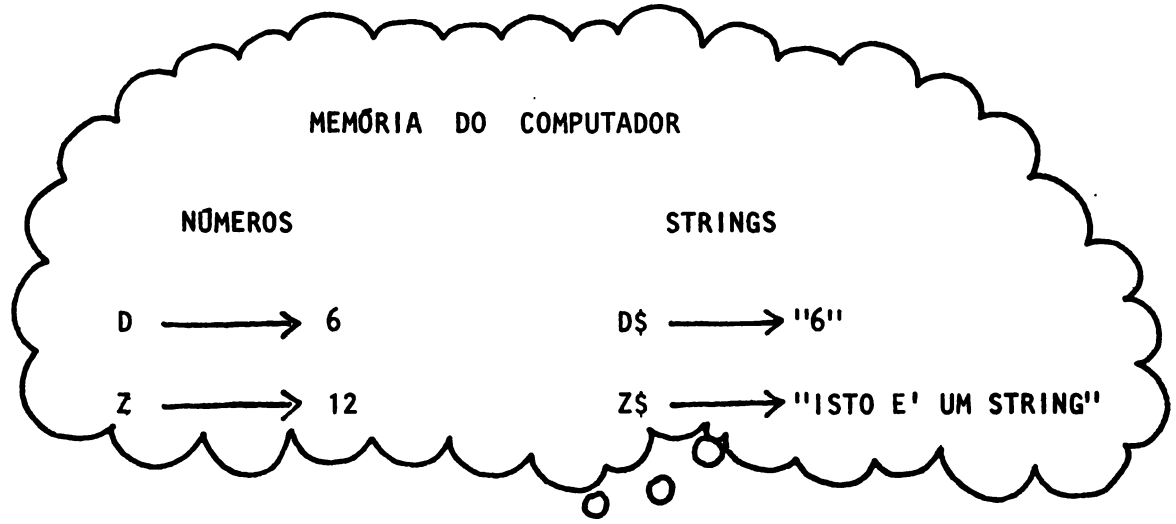
(1) Números sem aspas são dados numéricos.
(2) Dados numéricos só podem ser indicados por variáveis sem o sinal $ após o nome.

Digite isto, que o computador aceitará:

Não esqueça, siga sempre as regras do computador.

D = 6 (ENTER)
Z = 12 (ENTER)

Veja como ficou a memória do computador:



Agora você pode fazer coisas interessantes com estas letras. Digite:

O computador se lembra que D = 6.

PRINT D * 2 (ENTER)

O computador imprime o valor de D multiplicado por 2.

Tente esta linha:

PRINT Z/D

O computador imprime o resultado da divisão de Z por D.

Veja isto:

PRINT D$ * 2 (ENTER)

Tentou? Isto fez o computador imprimir ?TM ERROR. Porque ele não pode multiplicar um string por um número.

Risque os comandos que o computador rejeitará:

Não deixe de fazer este exercício

```
EXERCÍCIO COM VARIÁVEIS

F  = 22.9999999
M  = "19.2"
DZ$ = "LEMBRE-ME DISTO"
M$ = 15
Z  = F + F
```

Acabou? O computador aceitará as seguintes linhas:

```
F  = 22.9999999
DZ$ = "LEMBRE-ME DISTO"
Z  = F + F
```

REGRAS PARA VARIÁVEIS

Você pode usar dois caracteres para identificar uma variável. O primeiro caráter tem que ser uma letra de A a Z; o segundo pode ser uma letra ou um número. Se você quer colocar um string numa variável, ponha o sinal $ logo após o nome da mesma e caso deseje colocar um número, omita o sinal $.

ASSUNTOS APRENDIDOS NO CAPÍTULO 2

CONCEITOS

variáveis strings X variáveis numéricas

Agora que você aprendeu como o computador pensa, será fácil escrever alguns programas. Antes de irmos para o próximo capítulo, o que acha de uma parada para descanso? Relaxe...

NOTAS

CAPÍTULO 3

VEJA COMO É FÁCIL

3

VEJA COMO É FÁCIL

Digite:

NEW (ENTER)

Este comando apaga qualquer coisa que estiver na memória do computador.

Agora digite a linha abaixo com cuidado para não esquecer o número 10 antes do PRINT, ele é muito importante:

10 PRINT "OLA,SOU O SEU COMPUTADOR" (ENTER)

Pressionou a tecla (ENTER)? Nada aconteceu, não é? Nada que você possa ver. O que você fez foi digitar o seu primeiro programa. Digite:

RUN (ENTER)

O computador executa o programa.

Digite RUN (ENTER) outra vez e outra vez. Para sua satisfação a máquina mágica executa seu programa tantas vezes quantas desejar.

Já que está funcionando bem, vamos acrescentar outra linha ao programa.

Digite:

```
20 PRINT "QUAL E' O SEU NOME?"
```

Agora digite:

```
LIST (ENTER)
```

O computador mostra o seu programa. A tela ficará assim:

```
10 PRINT "OLA,SOU O SEU COMPUTADOR"
20 PRINT "QUAL E' O SEU NOME?"
```

O que deve acontecer quando você executar isto? Tente.

Digite:

```
RUN (ENTER)
```

O computador imprime:

```
OLA,SOU O SEU COMPUTADOR
QUAL E' O SEU NOME?
```

Responda à pergunta do computador e pressione (ENTER).

O quê? Ah...! Aquele chato do SN ERROR?

O computador não entendeu o que você quis dizer quando digitou seu nome. De fato, o computador não pode entender nada, a menos que você converse com ele em sua própria linguagem.

Se você fizer um erro e já tiver pressionado a tecla (ENTER), simplesmente redigite a linha.

Então vamos usar a palavra que o computador entende, INPUT.

Digite:

```
30 INPUT  A$  (ENTER)
```

Isto dirá ao computador para parar e esperar que você digite alguma coisa, que ele chamará de A$. Acrescente uma outra linha ao programa:

```
40 PRINT "OLA, " A$  (ENTER)
```

Agora execute um LIST para fazer uma comparação entre os programas. Digite:

```
LIST  (ENTER)
```

LEMBRE-SE: Se cometer um erro de digitação, redigite a linha.

Seu programa ficará assim:

```
10 PRINT "OLA,SOU O SEU COMPUTADOR"
20 PRINT "QUAL E' O SEU NOME?"
30 INPUT A$
40 PRINT "OLA, " A$
```

Você pode imaginar o que acontecerá quando executar este programa? Tente:

RUN (ENTER)

Ficou bom, não ficou? Veja abaixo o resultado após a execução de seu programa. (Caso tenha usado o mesmo nome)

```
OLA,SOU O SEU COMPUTADOR
QUAL E' O SEU NOME?
?MARIA
OLA, MARIA
```

Execute o programa outra vez com nomes diferentes:

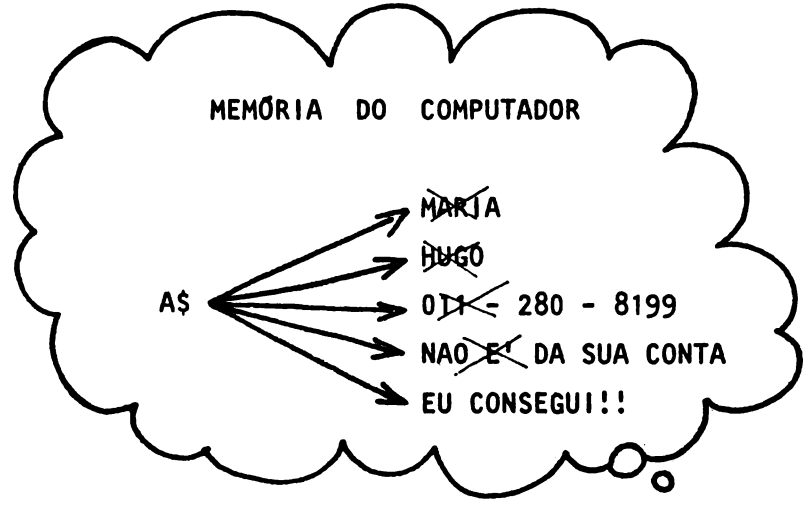
```
OLA,SOU O SEU COMPUTADOR
QUAL E' O SEU NOME?
?HUGO
OLA, HUGO
```

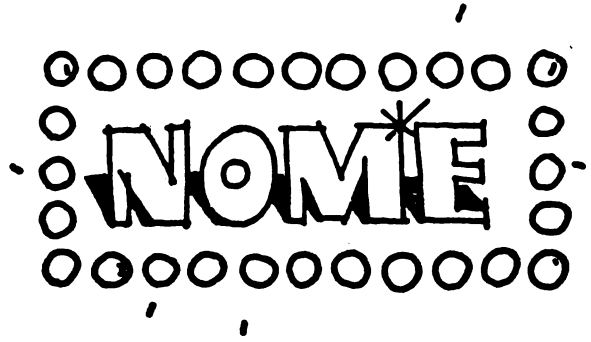
```
OLA,SOU O SEU COMPUTADOR
QUAL E' O SEU NOME?
?NAO E' DA SUA CONTA
OLA, NAO E' DA SUA CONTA
```

```
OLA,SOU O SEU COMPUTADOR
QUAL E' O SEU NOME?
?011 - 280 - 8199
OLA, 011 - 280 - 8199
```

```
OLA,SOU O SEU COMPUTADOR
QUAL E' O SEU NOME?
?EU CONSEGUI!!
OLA, EU CONSEGUI!!
```

Ao computador não interessa como você se chama. Acima exemplificamos o que você poderá informar ao computador na linha 30 (INPUT A$) cada vez que for executar o programa.





Existe uma forma mais fácil para você executar o programa quantas vezes desejar, sem digitar o comando RUN.

Digite esta linha:

```
50 GOTO 10
```

Agora execute. O programa será executado várias vezes sem parar.

GOTO disse ao computador para voltar para a linha 10:

```
10 PRINT "OLA,SOU O SEU COMPUTADOR"
20 PRINT "QUAL E' O SEU NOME?"
30 INPUT A$
40 PRINT "OLA, " A$
50 GOTO 10
```

Este programa não pára nunca, porque sempre que passa pela linha 50 (GOTO 10) ele volta para a linha 10.

Chamamos a isto de "LOOP". O único modo de parar um "LOOP" é pressionar a tecla (BREAK).

EVIDENCIE SEU NOME

Para apagar uma linha de um programa, digite o número da linha e pressione (ENTER). *Por exemplo:*

50 (ENTER)

Apaga a linha 50 do programa.

Mude a linha 50 e então dê a seu nome o tipo de atenção que ele merece. Como fazer isto? Digite:

```
50 GOTO 40
```

Seu programa ficará assim:

```
10 PRINT "OLA,SOU O SEU COMPUTADOR"
20 PRINT "QUAL E' O SEU NOME?"
30 INPUT A$
40 PRINT "OLA, " A$
50 GOTO 40
```

Digite RUN e veja o que o "LOOP" faz. Pressione (BREAK) quando estiver satisfeito.

Neste momento estamos abandonando o "OLA".

Há uma grande mudança que pode ser feita no comando PRINT. Simplesmente, coloque vírgula (,) ou ponto e vírgula (;). Primeiro tente a vírgula. Digite a linha 40 outra vez, mas com a vírgula no final:

```
40 PRINT A$,
```

Execute o programa. A vírgula força a impressão dos resultados em duas colunas.

Pressione (BREAK) e use o ponto e vírgula.

```
40 PRINT A$;
```

Faça outros testes usando vírgula e ponto e vírgula no comando PRINT.

Execute o programa. Você provavelmente não será capaz de dizer o que está acontecendo até pressionar (BREAK). Veja como o ponto e vírgula junta todos os resultados.

R E G R A S D E P O N T U A Ç Ã O

Veja como a pontuação após o PRINT faz o computador imprimir:

(1) VÍRGULA - Faz o computador imprimir em colunas.
(2) PONTO E VÍRGULA - Faz o computador imprimir tudo junto.
(3) SEM PONTUAÇÃO - Faz o computador imprimir em linha.

DEMONSTRAÇÃO DE COR E SOM

Neste programa estamos usando T como variável, mas poderíamos ter usado qualquer outra letra.

Vamos brincar um pouco mais com o computador e ver sua habilidade com cor e som. Primeiro limpe a memória. Lembra-se como? É claro, digite NEW.

Agora, digite este programa:

```
10 PRINT "PARA FAZER-ME MUDAR DE TOM"
20 PRINT "DIGITE UM NUMERO DE 1 A 255"
30 INPUT T
40 SOUND T, 50
50 GOTO 10
```

Veja como a linha 30 pergunta por T, e não T$. Isto acontece porque queremos dados numéricos e não strings.

Execute este programa para conseguir uma amostra de alguns sons do computador.

ERRO - Se você obtiver um ?FC ERROR quando executar o programa acima, é porque foi usado um número fora do intervalo de 1 a 255. Este erro, como todos os outros, faz com que o computador pare a execução do programa.

O que aconteceria se mudássemos a linha 40 para:

```
40 SOUND 50, T
```

Veja no capítulo 1, quando falamos sobre SOUND.

Entendeu isto? Quando fazemos tal mudança, o computador soa no mesmo tom o tempo todo,mas com uma duração diferente, dependendo do número que foi digitado.

Pressione (BREAK) primeiro e então apague este programa, digitando NEW. Agora veja se consegue escrever um programa semelhante, para fazer o computador apresentar a cor que você pedir. Lembre-se que existem 9 cores, com números de 0 a 8.

FAÇA VOCÊ MESMO O SEU PROGRAMA

OBS.: A linha 40 poderia ser: 40 CLS (T)

Este é o nosso programa:

```
10 PRINT "PARA FAZER-ME MUDAR DE COR"
20 PRINT "DIGITE UM NUMERO ENTRE 0 E 8"
30 INPUT T
40 CLS (T)
50 GOTO 10
```

ENSINE BOAS MANEIRAS AO SEU PROGRAMA

Programadores experientes vão achar que pressionar a tecla (BREAK) é uma maneira estúpida de parar um programa. Por que não tornamos o computador mais educado, fazendo-o perguntar se estamos prontos para terminar o programa? Mude a linha 50 para:

Pressione (BREAK) antes de digitar a linha.

```
50 PRINT "VOCE DESEJA VER OUTRA COR?"
```

acrescente estas linhas:

```
60 INPUT R$
70 IF R$ = "SIM" THEN 20
```

e execute o programa. Digite SIM e o programa será executado mais uma vez. Digite qualquer coisa diferente de SIM e o programa terminará.

O programa ficará assim:


```
10 PRINT "PARA FAZER-ME MUDAR DE COR"
20 PRINT "DIGITE UM NUMERO ENTRE 0 E 8"
30 INPUT T
40 CLS (T)
50 PRINT "VOCE DESEJA VER OUTRA COR?"
60 INPUT R$
70 IF    R$ = "SIM" THEN 20
```

Não se preocupe agora com esta linha IF/THEN. Vamos ter um capítulo sobre isto mais à frente.

Analisemos o que estas novas linhas fazem:

A linha 50 simplesmente imprime a pergunta.

A linha 60 diz ao computador para parar e esperar sua resposta R$.

A linha 70 diz ao computador para voltar à linha 20 se (IF) e somente se sua resposta R$ for SIM. Caso não seja, o programa termina pois não há mais comandos.

Você deu um grande passo neste capítulo. Esperamos com isto estar aumentando o seu apetite para aprender mais e mais.

Não se preocupe se não entendeu tudo perfeitamente, até agora. Simplesmente aprecie o seu computador.

ASSUNTOS APRENDIDOS NO CAPÍTULO 3

COMANDOS BASIC	TECLADO	CONCEITOS
Caracteres NEW INPUT GOTO RUN PRINT, PRINT; LIST IF/THEN	(BREAK)	Como modificar e apagar uma linha de programa.

NOTAS

CAPÍTULO 4

ENSINE-O A CONTAR

4

ENSINE-O A CONTAR

Neste capítulo vamos fazer algumas experiências com os efeitos de som do computador. Para isto temos que primeiro ensiná-lo a contar.

O porquê disto ficará claro mais tarde.

Digite:

```
10 FOR X = 1 TO 10
20 PRINT "X = " X
30 NEXT X
40 PRINT "TERMINEI DE CONTAR"
```

Digite NEW (ENTER) antes de começar um novo programa.

Execute o programa.

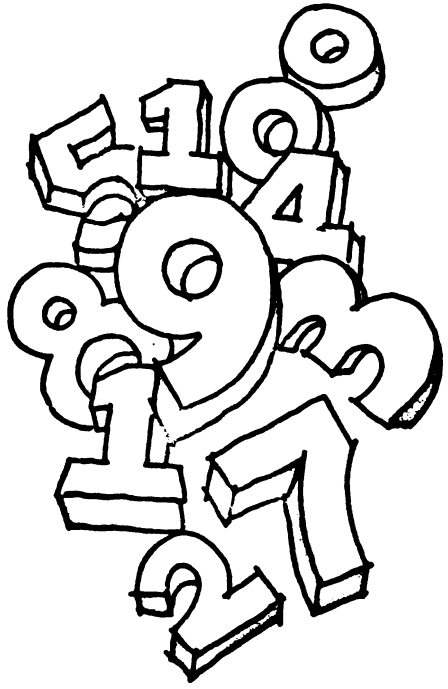
Execute este programa várias vezes, substituindo cada vez a linha 10 por uma das relacionadas a seguir:

```
10 FOR X = 1 TO 100
```

```
10 FOR X = 5 TO 15
10 FOR X = -2 TO 2
10 FOR X = 20 TO 24
```

Viu o que os comandos FOR e NEXT fizeram o computador executar? Eles o fizeram contar. Vamos estudar o último programa proposto:

```
10 FOR X = 20 TO 24
20 PRINT "X = " X
30 NEXT X
40 PRINT "TERMINEI DE CONTAR"
```



A linha 10 diz ao computador que o primeiro número deve ser 20 e o último 24. Ele utiliza a variável X para identificar estes números.

A linha 30 diz ao computador para voltar à linha 10 e contar o próximo número X, repetindo este ciclo até X alcançar o último número (24).

Dê uma olhada na linha 20. Como ela está entre o FOR e o NEXT, o computador deve imprimir o valor de X sempre que contar:

```
X = 20
X = 21
X = 22
X = 23
X = 24
```

Acrescente uma outra linha entre o FOR e o NEXT:

```
15 PRINT "... CONTANDO..."
```

Execute. Para cada cálculo, o computador executa todas as linhas colocadas entre o FOR e o NEXT.

Escreva um programa que faça o computador imprimir seu nome dez vezes.

FAÇA VOCÊ MESMO O SEU PROGRAMA (4/A)

OBS.: O programa deve contar até 10.

Escreva um programa para imprimir a tabela de multiplicação por 9 (9*1 até 9*10)

FAÇA VOCÊ MESMO O SEU PROGRAMA (4/B)

OBS.: PRINT 9*X é um comando válido.

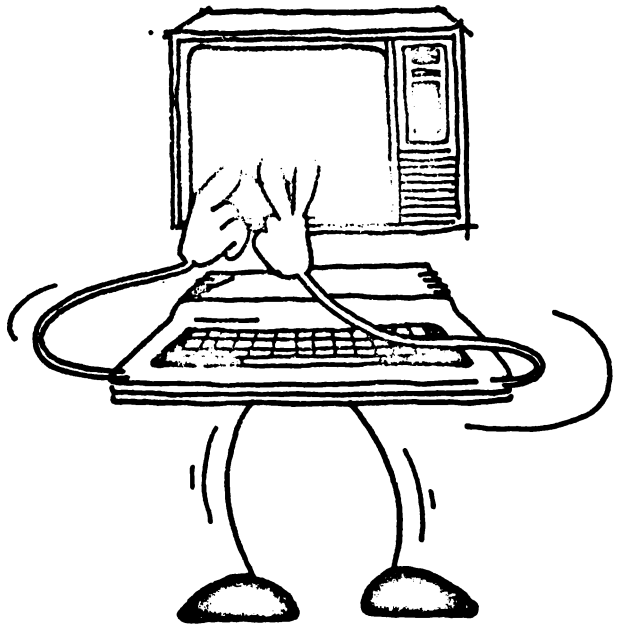
Escreva um programa para imprimir a tabela de multiplicação de 9*1 até 9*25.

FAÇA VOCÊ MESMO O SEU PROGRAMA (4/C)

OBS.: Acrescentando uma vírgula ao comando PRINT, você pode colocar todos os resultados numa única tela.

Acabou? Eis os nossos programas:

Programa 4/A	Programa 4/B	Programa 4/C
10 FOR X = 1 TO 10	10 FOR X = 1 TO 10	10 FOR X = 1 TO 25
20 PRINT "DENISE"	20 PRINT "9*"X"="9*X	20 PRINT "9*"X"="9*X
30 NEXT X	30 NEXT X	30 NEXT X



"2, 4, 6, 8,..."

CONTANDO DE DOIS EM DOIS

Agora, faremos o computador contar de uma forma diferente. Apague este programa (NEW) e digite o programa original, usando uma nova linha 10:

```
10 FOR X = 2 TO 10 STEP 2
20 PRINT "X = " X
30 NEXT X
40 PRINT "ACABEI DE CONTAR"
```

Execute o programa... Viu o que "STEP 2" fez? Ele conseguiu que o computador contasse de 2 em 2. A linha 10 diz ao computador que:

- O primeiro valor de X é 2
- O último valor de X é 10
- Deve-se adicionar 2 ao valor de X a cada passagem pelo NEXT X

Para fazer o computador contar de 3 em 3, faça assim:

```
10 FOR X = 3 TO 10 STEP 3
```

Execute o programa. Este será o resultado:

```
X = 3
X = 6
X = 9
```

Você deve estar pensando nos programas que executamos no início deste capítulo, onde não usávamos a palavra STEP. Se não a usarmos, o computador assume que o STEP é 1, isto é, adiciona 1 a cada NEXT.

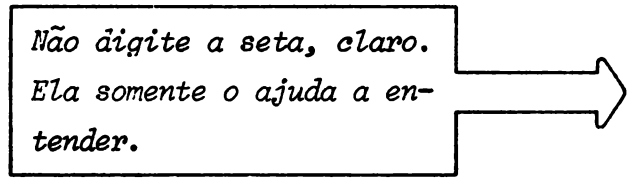
O computador não imprimiu o último X (10) porque 9 + 3 = 12 e 12 é maior que o limite máximo estipulado no comando FOR (10). Faça mais alguns exemplos com FOR... STEP e então verá mais claramente como eles funcionam.

```
10 FOR X = 5 TO 50 STEP  5
10 FOR X = 10 TO 1 STEP -1
10 FOR X = 1 TO 20 STEP  4
```

CONTANDO OS SONS

Agora que já ensinamos o computador a contar, podemos acrescentar algum som. Apague tudo com NEW e digite:

Não digite a seta, claro. Ela somente o ajuda a entender.



```
10 FOR X = 1 TO 255
20 PRINT "TOM" X
30 SOUND X, 1
40 NEXT X
```

Este programa faz o computador contar de 1 a 255 (de um em um). A cada número que conta, ele executa o que as linhas 20 e 30 mandam:

- Imprimir o valor atual de X na contagem (linha 20)
- Soar o tom correspondente a X (linha 30)

Por exemplo:

- Na primeira passagem pelo comando FOR, na linha 10, ele faz X igual a 1.

- Depois vai para a linha 20 e imprime 1, (o valor de X).

- Na linha 30 soa o tom correspondente ao nº 1.

- Na linha 40 ele volta para a linha 10 e faz X igual a 2.

- ... e assim por diante até X = 255

O que você acha que ocorrerá se a linha for alterada para:

```
10 FOR X = 255 TO 1 STEP -1
```

Tentou?

O computador soa tons do alto da seqüência para a base, soando cada nota.

Faça o computador soar tons:

1 - Da base da seqüência para o alto, soando a cada décima nota.
2 - Do alto da seqüência para a base, soando a cada décima nota.
3 - Do meio da seqüência para o alto, soando a cada quinta nota.

E X E R C Í C I O D E P R O G R A M A Ç Ã O
10 _____
10 _____
10 _____

Para parar temporariamente a execução de um programa, pressione (SHIFT) *e* (@) *simultaneamente. Para continuar, pressione qualquer tecla.*

Pronto para as respostas?

```
10 FOR X = 1 TO 255 STEP 10
10 FOR X = 255 TO 1 STEP -10
10 FOR X = 128 TO 255 STEP 5
```

Agora tente fazer um programa para o computador soar:

1 - Da base da seqüência para o alto e depois
2 - Do alto da seqüência para a base.

FAÇA VOCÊ MESMO O SEU PROGRAMA

A resposta encontra-se no apêndice I no fim deste manual.

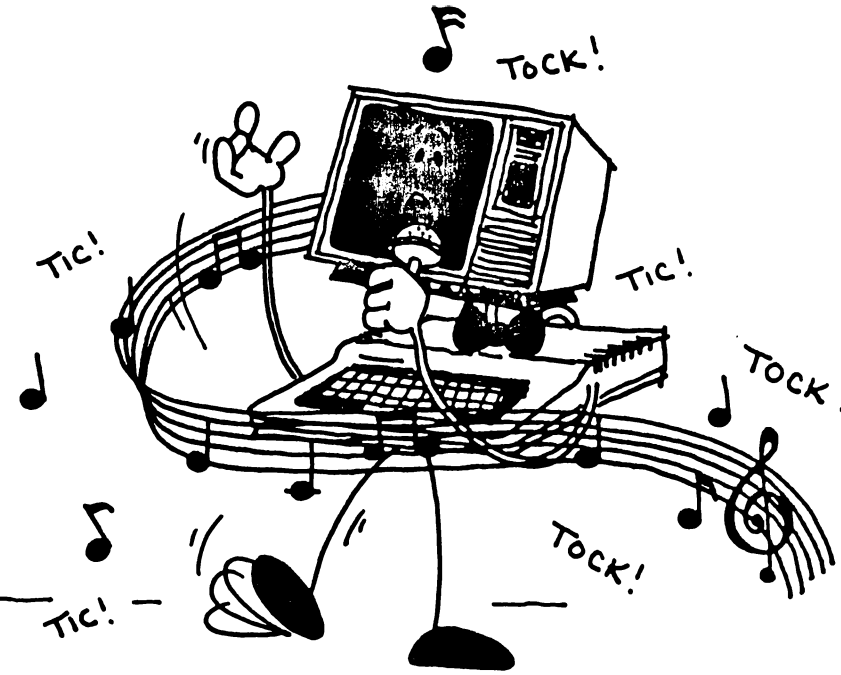
O COMPUTADOR PODE CANTAR ?

Sim. Embora o computador não tenha boa voz, ele pode cantarolar muitas canções. No próximo capítulo, mostraremos como ensinar-lhe algumas de suas canções favoritas.

ASSUNTOS APRENDIDOS NO CAPÍTULO 4

COMANDO BASIC	CARACTERES DO TECLADO
FOR... TO... STEP	(SHIFT) (@)
NEXT	

CAPÍTULO 5



CANTE A HORA

5

CANTE A HORA

Agora você está pronto para mostrar ao computador como fazer duas coisas: dizer a hora e cantar. (Bem... tanto quanto um computador pode cantar.) Como os dois ítens estão intimamente ligados, vamos apresentá-los num mesmo capítulo.

Comece digitando isto:

```
10 FOR Z = 1 TO 460 * 2
20 NEXT Z
30 PRINT "EU CONTEI ATE 920"
```

Execute o programa.

Seja paciente e espere alguns segundos. Dois segundos para sermos mais exatos, pois ele leva este tempo para contar de 1 a 920.

As linhas 10 e 20 criam uma pausa de tempo no programa, fazendo o computador contar até 920 e mantendo-o ocupado por dois segundos.

Como pode ver, isto nos dá uma idéia: vamos criar um cronômetro. Apague o programa e digite:

```
10 PRINT "QUANTOS SEGUNDOS VOU CONTAR?"
20 INPUT S
30 FOR Z = 1 TO 460 * S
40 NEXT Z
50 PRINT "CONTEI" S "SEGUNDOS"
```

Execute este programa, fornecendo o número de segundos que você deseja cronometrar.

Seria ótimo se o cronômetro soasse um tipo de alarme. Acrescente algumas linhas no fim do programa para fazer o computador soar um alarme.

FAÇA VOCÊ MESMO O SEU PROGRAMA

Aqui está o programa que fizemos:

```
10 PRINT "QUANTOS SEGUNDOS VOU CONTAR?"
20 INPUT S

30 FOR Z = 1 TO 460 * S
40 NEXT Z

50 PRINT "CONTEI" S "SEGUNDOS"

60 FOR T = 120 TO 180
70 SOUND T, 1
80 NEXT T

90 FOR T = 150 TO 140 STEP -1
100 SOUND T, 1
110 NEXT T

120 GOTO 50
```

É assim que os cronômetros computadorizados funcionam.

Observe a linha GOTO que colocamos no fim do programa. É o comando que mantém o alarme soando infinitamente, ou até que se pressione (BREAK) ou (SHIFT) (@) para parar o programa.

CONTANDO "DENTRO" DO TEMPO

Antes de irmos adiante com o relógio, vamos ensinar ao computador como contar "dentro" do tempo. O conceito lhe parecerá claro em poucos minutos.

Digite este programa:

Observe a vírgula logo após o PRINT. Tente sem a vírgula. A vírgula faz o computador imprimir :

"Y = " Y

na próxima coluna.

```
10 FOR X = 1 TO 3
20 PRINT "X = " X
30 FOR Y = 1 TO 2
40 PRINT, "Y = " Y
50 NEXT Y
60 NEXT X
```

Execute e veja como fica a tela:

```
X = 1
        Y = 1
        Y = 2
X = 2
        Y = 1
        Y = 2
X = 3
        Y = 1
        Y = 2
```

Chame isto uma contagem dentro de uma contagem ou um loop dentro de um loop, como prefe - rir.

Os programadores chamam isto de "ninho de loops". Eis a descrição do programa:

1 - Ele conta X de 1 a 3. Toda vez que ele incrementa X, ele:

A - Imprime o valor de X.

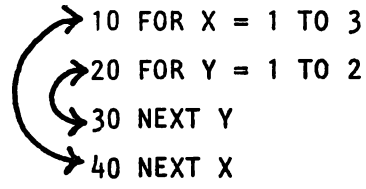
B - Conta Y de 1 a 2. Toda vez que ele incrementa Y, ele:

a - Imprime o valor de Y.

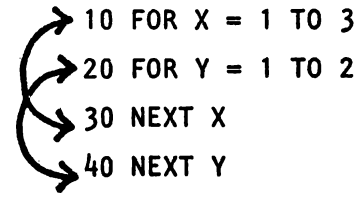
Quando você coloca um loop dentro de outro deve-se fechar o mais interno antes do mais externo.

Os loops nunca devem se cruzar.

CERTO

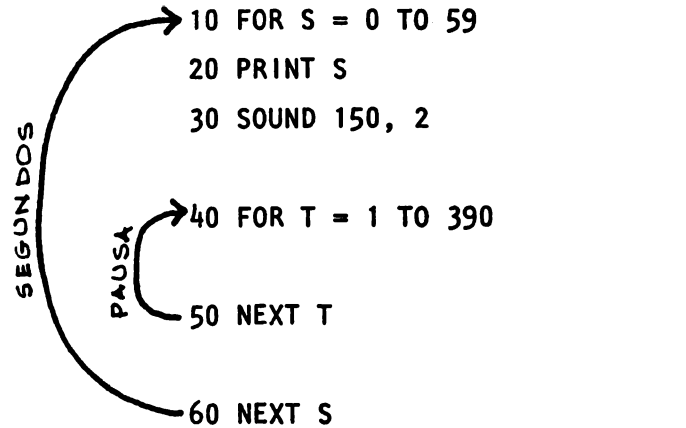


ERRADO



COMPARANDO ISTO A UM RELÓGIO

Com estas ferramentas, podemos exigir um pouco mais do computador. Digite:



Observe que alteramos o tempo da pausa na linha 40 para 390. (No programa anterior era 460). Por causa das mudanças efetuadas, tivemos que ajustar a pausa para um valor mais baixo.

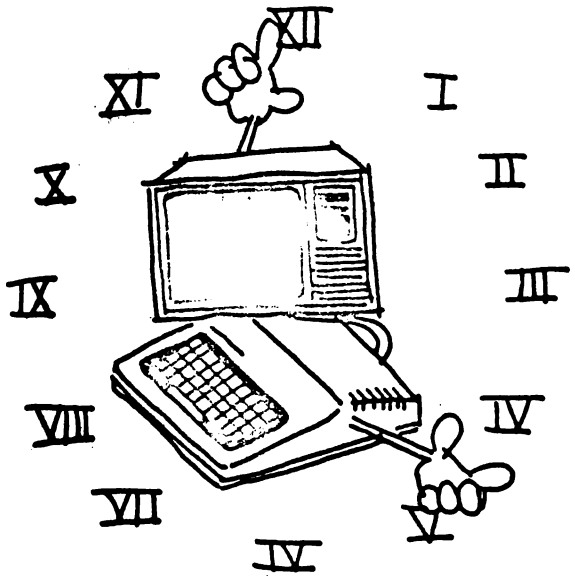
Execute o programa... Ele faz o seguinte:

1 - Conta os segundos de 0 a 59 e para cada segundo:

A - Imprime o segundo.

B - Soa o tom.

C - Faz uma pausa suficientemente longa para passar um segundo.



2 - Quando termina de contar todos os segundos, imprime uma mensagem informando que já passou um minuto, (linha 70).

Há um modo de fazermos com que este programa fique melhor. Coloque esta linha para limpar a tela:

```
15 CLS
```

Agora execute o programa. Desta vez, o computador executa as seguintes etapas:

1 - Ele conta os segundos de 0 a 59 (linhas 10 e 60) e para cada segundo:

A - Limpa a tela (linha 15)

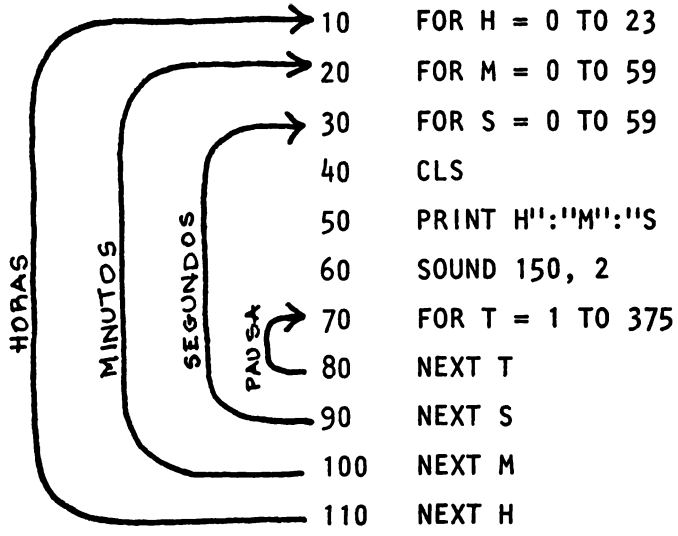
B - Imprime o segundo (linha 20)

C - Soa o tom (linha 30)

D - Faz uma pausa suficientemente longa para passar um segundo. (linhas 40 e 50)

2 - Quando termina de contar todos os segundos, imprime uma mensagem dizendo que já pas - sou um minuto, (linha 70).

Como este exemplo, é muito fácil fazermos um trabalho completo:



Aqui está um esboço do que o computador faz com este programa:

1 - Ele conta as horas de 0 a 23 (linha 10). Cada vez que conta uma nova hora:

A - Conta os minutos de 0 a 59 (linha 20). Cada vez que conta um novo minuto:

1 - Conta os segundos de 0 a 59. (linhas 30 e 90). Cada vez que conta um novo segundo:

a - Limpa a tela (linha 40).

b - Imprime a hora, minuto e segundo (linha 50).

c - Soa um tom (linha 60).

d - Faz uma pausa de tempo suficientemente longa para deixar passar um segundo (linhas 70 e 80).

Se acrescentarmos esta linha: 120 GOTO 10, o relógio funcionará infinitamente.

2 - Quando termina de contar todos os 59 segundos, ele volta à linha 20 para contar o próximo minuto (linha 100).

B - Quando termina de contar todos os 59 minutos, ele volta à linha 10 para contar a próxima hora (linha 110).

Achou este programa difícil? Pare um pouco e vá para o próximo. Mais tarde tudo parecerá bem mais fácil.

11- Quando termina de contar todas as horas (0 a 23), o programa termina.

Entre as linhas 90 e 100, podemos acrescentar um alarme, que soará a todo minuto. Escreva um programa que faça isto.

FAÇA VOCÊ MESMO O SEU PROGRAMA

Escreva um programa que faça o computador mostrar uma de suas 9 (nove) cores a cada segundo.

FAÇA VOCÊ MESMO O SEU PROGRAMA

As respostas dos dois programas encontram-se no apêndice I no fim deste manual.

PARA UM COMPUTADOR, ELE CANTA BEM!

Volte a ensinar ao computador a cantar. Vá ao apêndice A. Lá temos uma tabela de tons musicais que mostra o número correspondente a cada nota musical. Por exemplo, o tom número 89 corresponde ao DÓ médio (ou "C" na representação gráfica do apêndice A).

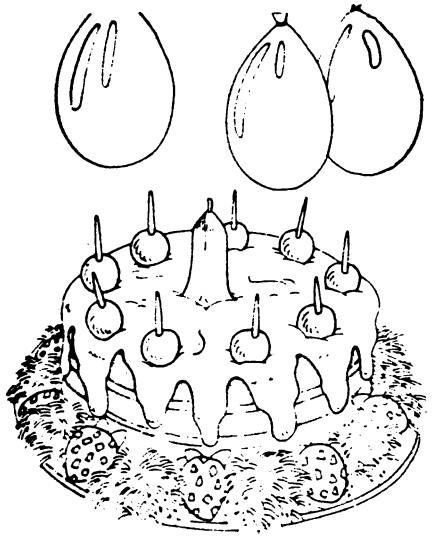
Infelizmente, o computador não pode soar fielmente todos os tons musicais. Por isso, ele canta um pouco desafinado. Mas, para aqueles que não têm mania de perfeição, até que atende muito bem.

Digite:

```
20 SOUND 89, 4
30 SOUND 89, 4
40 SOUND 108, 8
50 SOUND 89, 8
60 SOUND 133, 8
70 SOUND 125, 8
```

Execute o programa. São as seis primeiras notas de... Bem, você sabe, é uma música muito conhecida.

Para que as primeiras seis notas toquem outra vez, podemos usar um loop FOR/NEXT no programa.



Festa de aniversário.

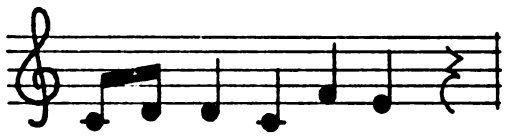
```
10 FOR X = 1 TO 2
20 SOUND 89, 4
30 SOUND 89, 4
40 SOUND 108, 8
50 SOUND 89, 4
60 SOUND 133, 8
70 SOUND 125, 8
80 NEXT X
```

Agora execute o programa novamente. Está faltando uma pausa, não é? É muito fácil resolver isto. Basta acrescentar estas linhas:

```
74 FOR Y = 1 TO 230
76 NEXT Y
```

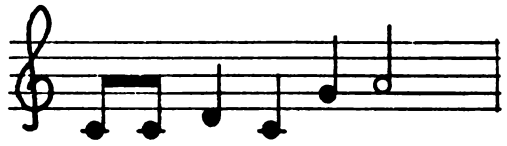
Execute outra vez. Agora começou a soar como uma coisa real, não é?

Aqui está um programa que ilustra as primeiras duas frases da canção:



Pa-ra-béns prá vo-cê

```
10 SOUND  89, 4        "Pa"
20 SOUND  89, 4        "ra"
30 SOUND 108, 8        "béns"
40 SOUND  89, 8        "prá"
50 SOUND 133, 8        "vo"
60 SOUND 125, 8        "cê"

64 FOR Y = 1 TO 230    (pausa)
66 NEXT Y
```



nes-ta da-ta que-ri-da

```
70 SOUND  89, 4        "Nes"
80 SOUND  89, 4        "ta"
90 SOUND 108, 8        "da"
100 SOUND  89, 8       "ta"
110 SOUND 147, 8       "que"
120 SOUND 133, 16      "rida"
```

O programa está ficando muito grande e não cabe todo na tela? Tente isto: LIST 10-64 (ENTER). Somente metade do programa será mostrado.

ASSUNTOS APRENDIDOS NO CAPÍTULO 5	
COMANDOS BASIC	CONCEITO DE PROGRAMAÇÃO
CLS	Ninho de loops

CAPÍTULO 6

DECISÕES,

DECISÕES...

6

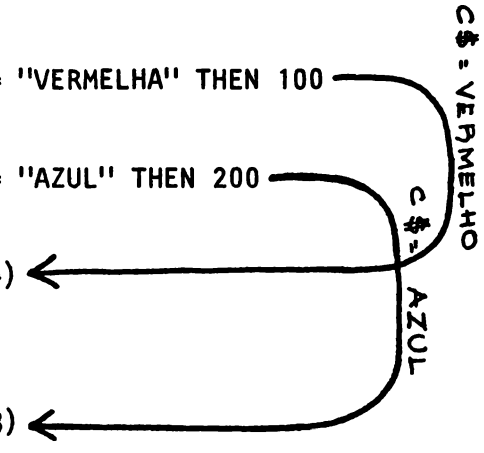
DECISÕES, DECISÕES...

Aqui está uma decisão fácil para o computador:

1 - Se (IF) digitarmos VERMELHA então (THEN) faça a tela ficar vermelha ou ...
2 - Se (IF) digitarmos AZUL então (THEN) faça a tela ficar azul.

É muito fácil? Deixemos o computador fazer isto. Digite este programa:

```
10 PRINT "VOCE QUER A TELA VERMELHA OU AZUL?"
20 INPUT C$

30 IF C$ = "VERMELHA" THEN 100

40 IF C$ = "AZUL" THEN 200

100 CLS (4)
110 END

200 CLS (3)
```



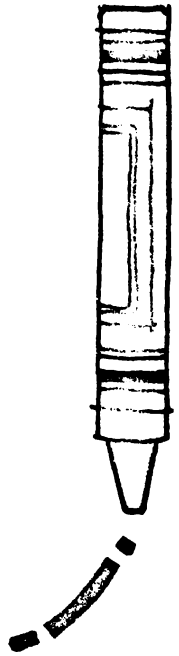
Não fique confuso por causa das setas ou dos espaços entre as linhas do programa. Eles servem para ilustrar o fluxo do programa e facilitar a sua compreensão.

Execute o programa várias vezes, digitando vermelha e azul. Vejamos o que o programa está fazendo:

Se (IF) você digitou VERMELHA... então (THEN)...

A linha 30 manda o programa para a linha 100. A linha 100 faz a tela ficar vermelha. Neste ponto temos que impedir o computador de ir à linha 200.

A linha 110 faz justamente isto. Ela termina o programa lá... Uma vez que atinja a linha 110, ele nunca chegará a linha 200.



... ou então ...

Se (IF) você digitar AZUL ... então (THEN) ...

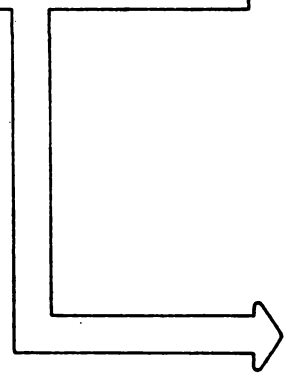
A linha 40 manda o computador para a linha 200, que faz a tela ficar azul. Não é preciso colocar END na próxima linha porque a linha 200 é a última linha do programa.

O que acontece se você digitar algo diferente de VERMELHA ou AZUL? Tente executar o programa, digitando VERDE em resposta à pergunta do computador.

Ele tornará a tela vermelha, certo? Sabe por quê?

Sendo a condição não verdadeira, o THEN é ignorado e o computador parte para a próxima linha do programa.

Coloque o número das linhas na parte pontilhada.



Há duas linhas que poderiam ser acrescentadas ao programa para fazer o computador repetir a pergunta caso a resposta não seja vermelha nem azul. Vamos fornecer as duas linhas e deixá-lo imaginar onde colocá-las no programa:

```
E X E R C Í C I O    D E    P R O G R A M A Ç Ã O


... PRINT "VOCE DEVE DIGITAR VERMELHA OU AZUL"

... GOTO 20
```

Observações:

(1) As linhas devem ser colocadas após o computador ter tido a chance de testar se sua resposta é vermelha ou azul.

(2) As linhas devem vir antes do computador colocar a cor vermelha na tela.

Você descobriu onde colocar as duas linhas no seu programa?

Elas devem vir após a linha 40 e antes da 100.

```
50 PRINT "VOCE DEVE DIGITAR VERMELHA OU AZUL"
60 GOTO 20
```

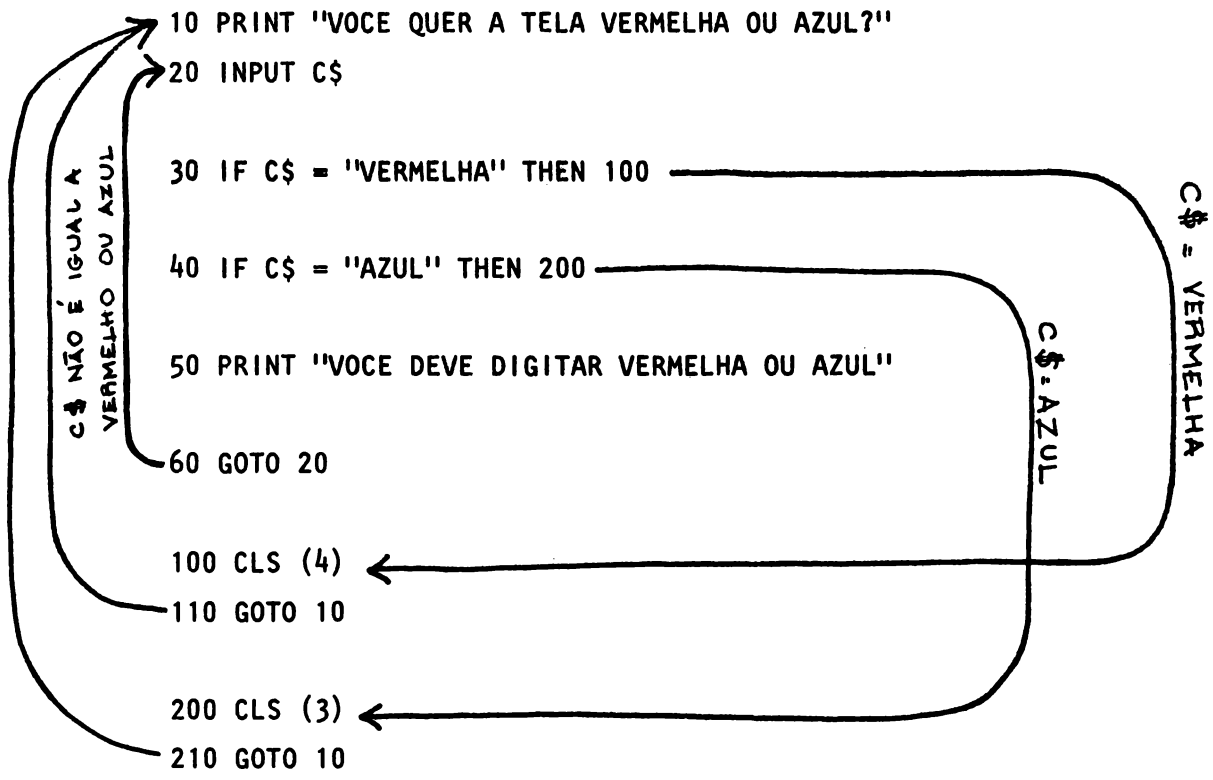
Veja se é possível fazer mais alguma alteração no programa.

Depois de tornar a tela azul ou vermelha, faça o computador, ao invés de encerrar o programa, voltar a solicitar a digitação de vermelha ou azul.

FAÇA VOCÊ MESMO O SEU PROGRAMA

OBS.: Você precisa trocar a linha 110 e acrescentar a linha 210.

Já escreveu o programa? Veja o nosso:



Para acompanhar os passos do computador neste programa, passe de uma linha para outra , seguindo as setas desenhadas no diagrama. Veja a diferença entre as setas das linhas IF/ THEN e GOTO:

R E G R A S D E I F / T H E N E G O T O

IF/THEN é um desvio condicional.

. Você só deve seguir estas setas se a condição (C$ = vermelha ou C$ = azul) for verdadeira.

GOTO é um desvio incondicional.

. Você segue estas setas sempre que chegar a uma linha de GOTO.

Embora este capítulo tenha sido curto, você aprendeu um dos conceitos de programação mais importantes.Você usará o computador para tomar decisões por todo o resto deste manual.

A S S U N T O S A P R E N D I D O S N O C A P Í T U L O 6

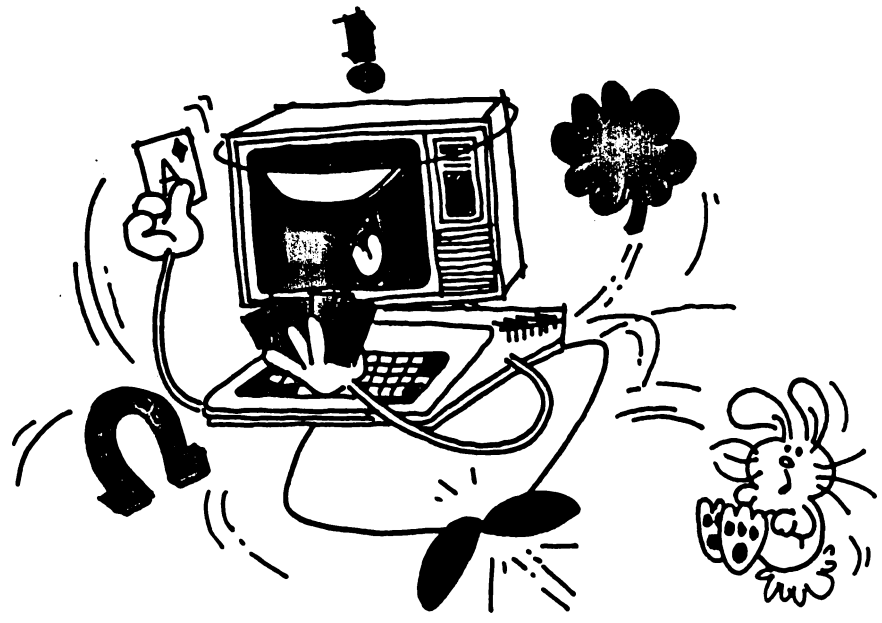
COMANDOS BASIC

IF / THEN

END

NOTAS

CAPÍTULO 7



JOGOS

7

JOGOS

Graças ao comando RND, o computador pode jogar qualquer tipo de jogo envolvendo a sorte . Mesmo que não planeje jogar com o computador, você vai querer saber como usar RND e PRINT @ . Vamos apresentá-los neste capítulo e também lhe mostraremos outros usos do IF/THEN.

Digite:

```
10 PRINT RND (10)
```

Execute. O computador imprime um número aleatório ou randômico de 1 a 10. Execute outras vezes...

É como se o computador estivesse sorteado um número de 1 a 10. É impossível prever o número que ele vai escolher. Digite e execute este programa. Pressione (BREAK) quando estiver convencido de que o computador está imprimindo números aleatórios:

Para fazer o computador parar temporariamente enquanto estiver executando o programa, pressione (SHIFT) *e* (@) *simultaneamente. Para continuar a execução do programa pressione qualquer tecla.*

```
10 PRINT RND(10)
20 GOTO 10
```

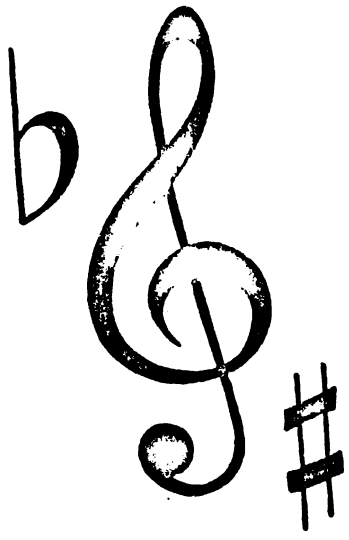
Para que o computador pegue números aleatórios de 1 a 100, mude a linha 10 para:

```
10 PRINT RND (100)
```

Como fará você para que o computador pegue, randomicamente, números de 1 a 255?

A resposta é:

```
10 PRINT RND (255)
```



UM COMPLETO SHOW RANDÔMICO

Apenas por brincadeira, vamos fazer o computador executar uma canção feita de tons randômicos. Digite:

```
10 T = RND (255)
20 SOUND T, 1
30 GOTO 10
```

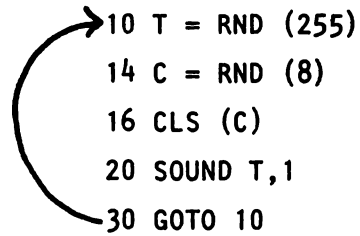
Execute. Grandiosa música, não? Pressione (BREAK) quando estiver satisfeito.

Para fazer uma bela apresentação visual randômica neste programa, coloque duas linhas de

forma que o computador mostre uma cor randômica antes de tocar qualquer tom randômico.

FAÇA VOCÊ MESMO O SEU PROGRAMA

Aqui está o que fizemos:

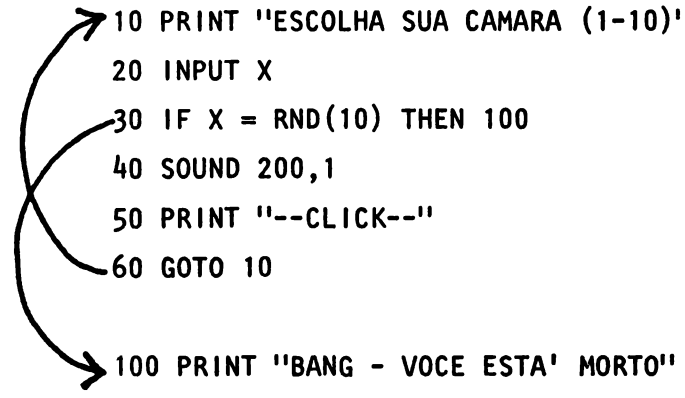


Vamos mostrar alguns jogos simples neste capítulo. Sinta-se livre, torne-os mais interessantes com sua imaginação ou crie, você mesmo,seus próprios jogos.

ROLETA RUSSA

Lembre-se: Digite sempre NEW (ENTER), antes de começar um novo programa.

Nesta roleta russa a arma tem dez câmaras. O computador escolhe, randomicamente, uma das 10. Somente uma está com a bala fatal. Digite:



Primeiro, na linha 20, o jogador fornece X, um número de 1 a 10. Então, o computador compara X com RND (10), um número escolhido aleatoriamente de 1 a 10, pelo computador.

Agora observe as setas que desenhamos.

Se X não é igual a RND (10), então o computador faz "click" e volta à linha 10 onde lhe é dado uma nova chance.

Vamos fazer uma rotina de morte melhor a partir da linha 100.

Digite:

Você ainda se lembra como mostrar uma parte do programa?

LIST 50-130

Mostra a parte do meio do programa.

```
100 FOR T = 133 TO 1 STEP - 5
110 PRINT "BANG!!!"
120 SOUND T,1
130 NEXT T
140 CLS
150 PRINT @  226, "DESCULPE-ME, VOCE ESTA' MORTO"
160 SOUND 1,50
170 PRINT @  290, "PROXIMA VITIMA, POR FAVOR"
```

Execute o programa. Eis o que acontece:

As linhas 100 até 130 fazem o computador produzir um som de tom descendente e imprimir BANG!!! muitas vezes na tela.

Isto lhe ajudará a ver um programa grande. Pressione (SHIFT) e (@) quando o computador começar a mostrar o programa na tela, assim ele pára a listagem. Para continuar pressione qualquer tecla.

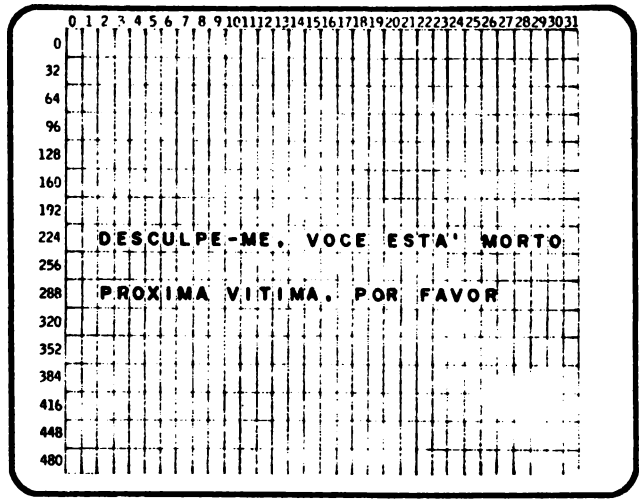
A linha 140 limpa a tela e como não indicamos a cor da tela, o computador assume o verde.

Olhe as linhas 150 e 170. Ambas usam PRINT @ . Veja como isto funciona.

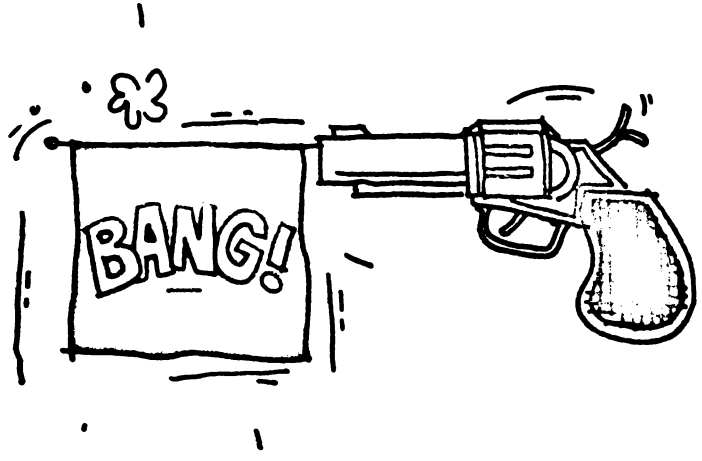
Observe a matriz que temos na página seguinte, mostrando cada uma das 512 posições da tela do seu vídeo. Quando fizemos o programa, digitamos estas duas mensagens: "DESCULPE-ME, VOCE ESTA' MORTO" e "PROXIMA VITIMA, POR FAVOR" na matriz, posicionando-as onde as queríamos na tela.

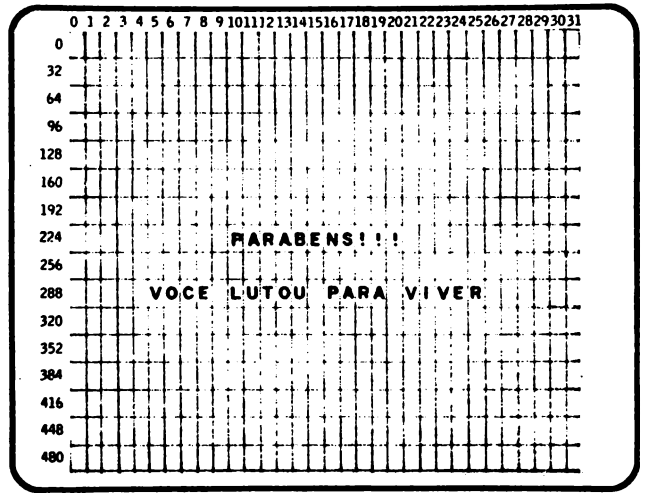
DESCULPE-ME, VOCE ESTA' MORTO começa na posição 226(224+2); PROXIMA VITIMA, POR FAVOR, começa na posição 290(288+2). Usando estes números na linha PRINT @ , simplesmente dizemos ao computador onde queremos imprimir a mensagem.

Esta matriz foi colocada no apêndice C deste manual. Use-as nos programas. Uma boa formatação de tela pode aumentar o interesse pelos seus programas

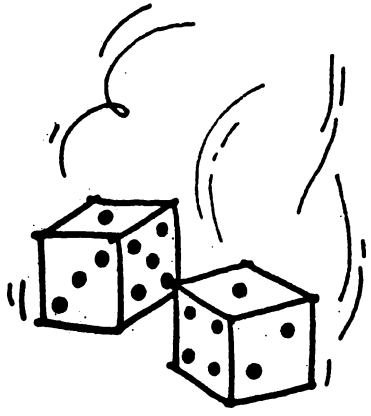


Mude este programa. Se o jogador conseguir ficar vivo durante "10 clicks", o computador elege-o vencedor e imprime a seguinte mensagem na tela:





FAÇA VOCÊ MESMO O SEU PROGRAMA



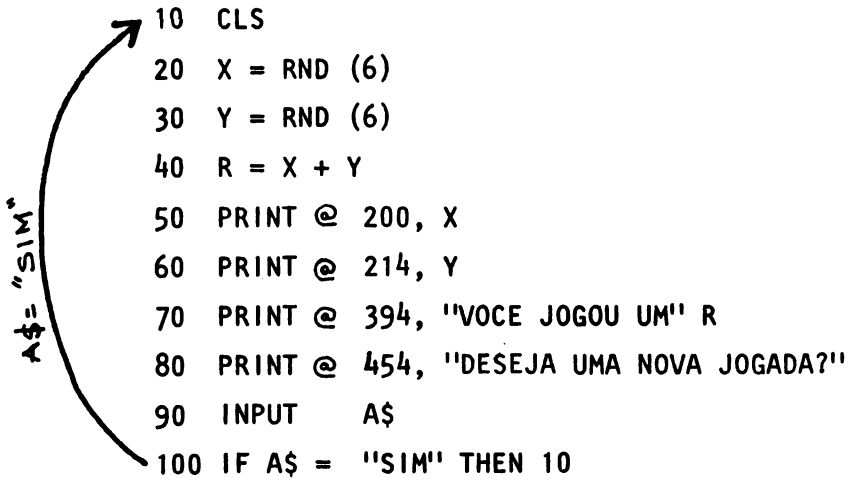
"Perdedor"

OBS.: Você pode usar o loop FOR/NEXT para que o computador conte o número de clicks.

Nossa resposta está no apêndice I no fim deste manual.

JOGANDO DADOS

Para brincarmos com o nosso próximo jogo, temos que ensinar ao computador como jogar dados. Para isto, ele precisa jogar dois dados, isto é, deve fornecer dois números randômicos. Digite:



Execute o programa. Vamos dar uma olhada.

A linha 10 diz ao computador para limpar a tela.

A linha 20 faz o computador pegar um número randômico de 1 a 6 para um dos dados.

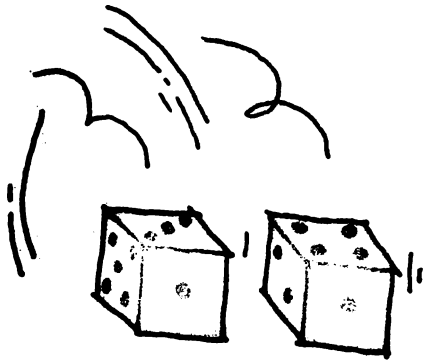
A linha 30 faz o computador pegar um número randômico de 1 a 6 para outro dado.

A linha 40 soma os números dos dois dados para obter o valor total da jogada.

As linhas 50, 60 e 70 imprimem o resultado da jogada na tela.

Na linha 90, pode-se fazer uma nova jogada. Se digitar "SIM", o computador volta à linha 10 e executa o programa outra vez. Caso contrário, sendo esta a última linha do programa, ele pára.

CRAPS



"Vencedor"

Agora que você já sabe como o computador joga dados, será muito fácil preparar um programa de "CRAPS". Estas são as regras do jogo:

1 - O jogador joga os dois dados e se conseguir um total de 2, 3 ou 12 pontos na primeira jogada, ele ganha e o jogo termina.

2 - Se o jogador consegue 7 ou 11 na primeira tentativa, ele perde e o jogo acaba.

3 - Se for obtido qualquer outro número na primeira rodada, este número se transforma no "ponto" do jogador e ele deve continuar jogando até repetir o valor do ponto e ganhar ou então marcar 7 e perder o jogo.

Você já sabe mais do que o suficiente para escrever este programa. Faça-o.

Mande o computador imprimir as informações de uma forma bonita na tela e mantenha o jogador informado sobre o que está ocorrendo. Você poderá demorar um pouco para terminar, mas faça o melhor que puder. Boa sorte!

FAÇA VOCÊ MESMO O SEU PROGRAMA

OBS.: Nossa resposta está no apêndice I no fim deste manual.

ASSUNTOS APRENDIDOS NO CAPÍTULO 7

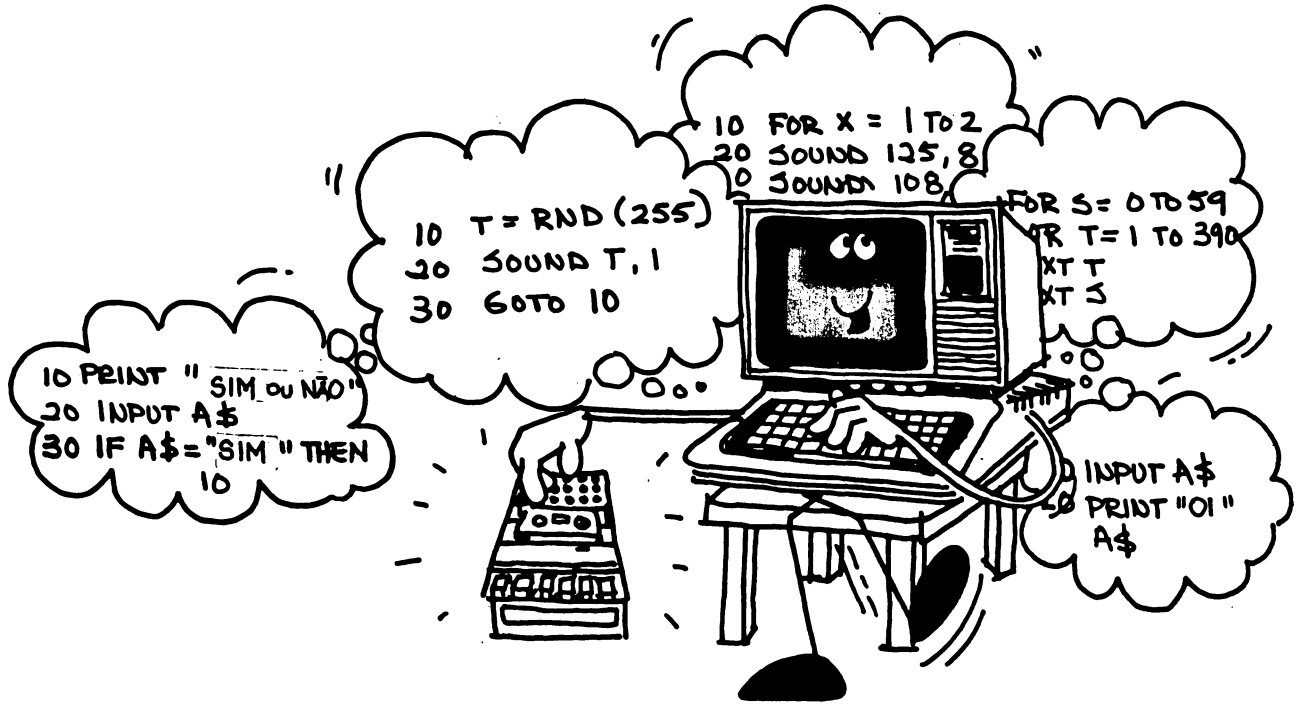
COMANDOS BASIC

RND

PRINT @

NOTAS

CAPÍTULO 8



GRAVE EM FITA

8

GRAVE EM FITA

Logo você estará escrevendo longos e complicados programas. Talvez até já esteja e imaginamos, que deve se aborrecer bastante ao ver seu programa desaparecer sempre que a máquina é desligada.

É possível salvar qualquer programa (guardar uma cópia), em fita cassete. Uma vez guardado você pode carregar o programa de volta à memória do computador sempre que quiser.

Este capítulo é apenas para aqueles que possuem um gravador cassete e queiram usá-lo. Se não possuir um, pule para o próximo capítulo ou corra à primeira loja e adquira um.

COMO SALVAR UM PROGRAMA

1 - Digite um programa no computador. Execute-o para ver se está correto.

2 - Rebobine a fita cassete e pressione os botões PLAY e RECORD ao mesmo tempo, até que fiquem bloqueados.

3 - Dê um nome ao programa que quer salvar. Pode-se usar qualquer nome com oito letras ou menos. No exemplo, vamos usar "NOME".

4 - Salve-o na fita, digitando este comando:

CSAVE "NOME" (ENTER)

Assim que o motor do gravador começar a funcionar você estará gravando o programa na fita. Olhe a tela. Quando o computador imprimir:

OK

e o motor parar, seu programa estará gravado na fita. Ele continua também na memória, pois foi apenas copiado.

CARREGANDO UM PROGRAMA

Inverter o processo e carregar o programa da fita para o computador, também é muito fácil:

1 - Certifique-se de que a fita está toda rebobinada.

2 - Pressione o botão PLAY até bloqueá-lo. Regule o nível de volume do gravador.

3 - Digite NEW para limpar a memória.

4 - Digite CLOAD com o nome de seu programa como segue:

CLOAD "NOME" (ENTER)

Se existirem vários programas na fita, o computador imprimirá o nome de cada programa que achar antes daquele que você quer carregar.

O motor do gravador dá partida. Olhe para a tela, no seu canto superior esquerdo aparecerá a letra:

S

Isto significa que o computador está procurando o programa. Quando for achado, ele imprime a letra F e o nome do programa. Por exemplo, se o nome do programa for "NOME", vai aparecer no alto da tela:

F NOME

Se você tentar carregar um programa que não está na fita, o computador não vai parar de procurá-lo . Pressione RESET para interromper a busca.

Quando o computador imprimir:

OK

e o motor do gravador parar, o programa estará na memória, pronto para ser executado.

SALVANDO MAIS DE UM PROGRAMA

Para salvar mais de um programa na mesma fita, você deve certificar-se de não estar gravando sobre outro programa. A seguir apresentamos a maneira mais fácil de posicionar a fita no fim de seu último programa:

1 - Rebobine a fita.

2 - Pressione o botão PLAY.

3 - Digite SKIPF e o nome do último programa na fita. Por exemplo, se o último programa for "NOME", digite:

```
SKIPF "NOME"
```

Quando for alcançado o final do programa "NOME", o motor do gravador pára e aparece na tela:

```
OK
```

4 - Uma vez posicionada a fita no fim do último programa, pressione RECORD e PLAY, dê um nome ao programa e salve-o.

Se você esqueceu o nome do último programa, digite:

Você pode substituir o nome X por qualquer outro que não esteja na fita.

SKIPF "X"

e observe a tela. O computador indicará o nome de cada programa que encontrar na fita e imprimirá a mensagem: I/O ERROR quando chegar ao final da fita. Mas não se preocupe com isto pois você encontrou o que estava procurando: o nome do último programa na fita.

Agora você pode digitar o comando SKIPF com o nome do último programa. Não se esqueça de rebobinar a fita.

MACETES PARA FAZER BOAS GRAVAÇÕES

- Quando não estiver usando o gravador para salvar ou carregar programas, não deixe as teclas RECORD e PLAY bloqueadas. Pressione STOP.

- Não tente gravar numa fita gravada anteriormente pelo computador, porque embora o processo apague a gravação anterior, muitas informações antigas podem se confudir com as novas. Caso queira usar a mesma fita duas ou três vezes, use um aparelho apropriado para apagar fitas (bulk tape eraser).

Faça agora novas gravações para testar o comando CSAVE. Não esqueça de exercitar o comando CLOAD também.

ASSUNTOS APRENDIDOS NO CAPÍTULO 8

COMANDOS BASIC

CLOAD

CSAVE

SKIPF

NOTAS

CAPÍTULO 9

COLOQUE COR NA SUA TELA

9

COLOQUE COR NA SUA TELA

Você já aprendeu bastante e pode começar a usar cores. Como a vontade de fazer gráficos coloridos vem rapidamente, e bons programas gráficos são difíceis de escrever, este capítulo vai mostrar como começar. Ao longo do capítulo você sentirá, certamente, vontade de acrescentar novas instruções aos nossos programas, ou até mesmo criar seus próprios. Esperamos que faça isto, é uma maneira rápida de aprender.

Para começar, digite:

```
10 CLS (0)
```

Para fazer a tela ficar preta, acrescente estas duas linhas e execute o programa:

```
20 SET (0,0,4)
30 GOTO 30
```

Certifique-se de que, realmente digitou a linha 30. Vamos explicar o porquê mais tarde.

Você está vendo o ponto vermelho? Ele está no alto à esquerda da tela. Para colocá-lo no

canto direito, mude a linha 20 e execute o programa:

```
20 SET (63,31,4)
```

Caso deseje colocar o ponto no meio da tela, execute o programa usando a linha 20 a se - guir:

```
20 SET (31,14,4)
```

SET diz ao computador para colocar um ponto na tela, numa certa posição horizontal e vertical.

- O primeiro número digitado é a posição horizontal e deve ser um número entre 0 e 63.

- O segundo número é a posição vertical e deve ser um número entre 0 a 31.

No apêndice D há uma matriz de tela "Posicionamento de gráficos". A matriz divide a tela nas 64 (0 a 63) posições horizontais e 32 (0 a 31) posições verticais. Use esta matriz para desenhar os gráficos.

Tudo isto explica o significado dos dois primeiros números. Mas, e o terceiro? Tente usar outros números que não seja o 4, como terceiro número. Digite estas linhas e execute o programa:

```
20 SET (31,14,3)
20 SET (31,14,1)
```

A posição na tela é indicada de forma diferente nos comandos SET e PRINT @ . Por isso, existem três matrizes nos apêndices. Cuidado para não usar a matriz errada.

Entendeu? Com o número 3, aparece um ponto azul e com o número 1 um ponto verde. Os códigos são os mesmos adotados para o comando CLS, 0 a 8. Eles estão relacionados no apêndice B (cores e caracteres gráficos).

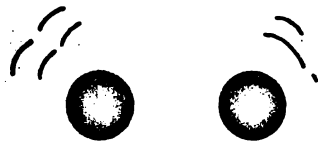
Agora, qual o objetivo da linha GOTO? Apague a linha GOTO e execute o programa:

```
10 CLS
20 SET (31,14,1)
```

Parece que o ponto não foi impresso. Realmente ele foi, mas quando o programa acabou, o computador imprimiu OK em cima dele.

Para evitar esta situação, digite GOTO no fim do programa. Ele fará um loop infinito, de maneira que o programa não parará nunca e o ponto continuará impresso, indefinidamente.

POSICIONANDO DOIS PONTOS



"Coloque o ponto!"

Para posicionar mais de um ponto, é preciso fazer um pequeno planejamento. Apague todo o programa e execute este:

```
10 CLS (0)
20 SET (32,14,3)
30 SET (33,14,3)
40 GOTO 40
```

Devem existir, agora, dois pontos azuis, lado a lado, no meio da tela.

Agora mude a cor do ponto da direita de modo a ter um ponto azul e um vermelho. Digite:

```
30 SET (33,14,4)
```

E execute o programa... ambos os pontos são vermelhos. Por que? Vejamos:

Olhe outra vez a matriz (Posicionamento de gráficos) no apêndice D. Observe as linhas mais escuras do gráfico. Os pontos estão agrupados 4 a 4 entre linhas mais escuras.

Por exemplo, o bloco do meio da matriz contém estes 4 pontos:

	HORIZONTAL	VERTICAL
Posição	32	14
Posição	33	14
Posição	32	15
Posição	33	15

Cada ponto dentro do bloco deve ser:

1 - Da mesma cor (cores 1-8)

ou

2 - Preto

No nosso programa, tentamos fazer o computador mostrar dois pontos com cores diferentes , azul e vermelho, dentro do mesmo bloco. Como o computador não pode fazer isto, ele assume que os dois pontos são da segunda cor, vermelha.

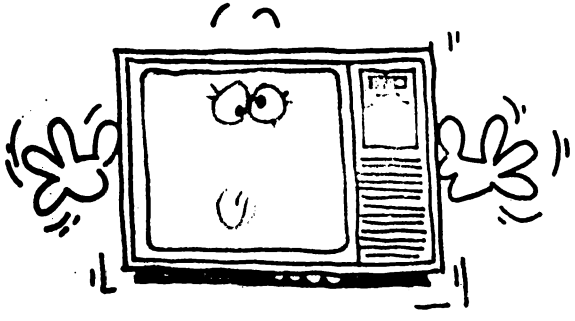
Digite isto e execute o programa:

```
30 SET (34,14,4)
```

Como o ponto na posição 34,14 está em um bloco diferente do outro, o computador pode mostrar os dois pontos de cores diferentes.

A CARA DO COMPUTADOR

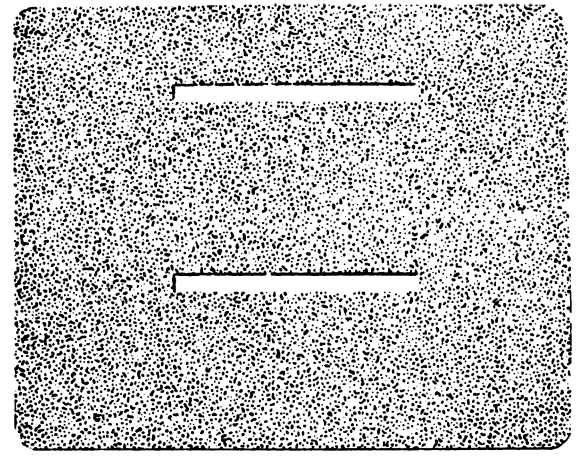
Com estas explicações, você pode, agora, desenhar o que quiser. Vamos desenhar simplesmente a cara do computador. Primeiro desenhe o alto e a parte de baixo da cabeça. Digite:



"Que cara engraçada!"

```
5  CLS (0)
10 FOR H = 15 TO 48
20 SET (H,5,5)
30 SET (H,20,5)
40 NEXT H
50 GOTO 50
```

Isto é o que você deve ter na tela:



As linhas 10 e 40 fazem um loop, FOR/NEXT para variar H das posições horizontais de 15 a 48 nos traços de cima e de baixo, respectivamente.

A linha 20 define o traço de cima. As posições horizontais vão de 15 a 48 e a posição vertical é 5.

A linha 30 define o traço de baixo. As posições horizontais vão novamente de 15 a 48, sendo a posição vertical 20.

Para traçar o lado direito e esquerdo da cabeça, digite estas linhas:

Note que trocamos a linha 50 (a linha do GOTO).

```
50 FOR V = 5 TO 20
60 SET (15,V,5)
70 SET (48,V,5)
80 NEXT V
90 GOTO 90
```

e execute.

Vamos fazer o nariz laranja. Digite:

```
90 SET (32,13,8)
```

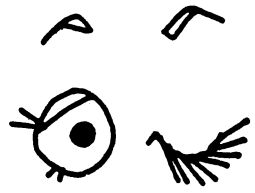
e a boca vermelha. Digite:

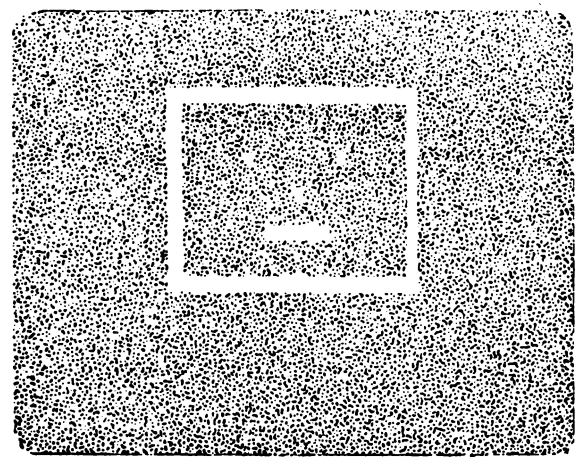
```
100 FOR H = 28 TO 36
110 SET (H,16,4)
120 NEXT H
```

e os olhos azuis. Digite:

```
130 SET (25,10,3)
140 SET (38,10,3)
150 GOTO 150
```

e execute o programa. Isto é o que você verá na tela:





UMA PISCADA DE OLHO

Acrescentando duas linhas, você fará o computador "piscar" o olho.

Você não precisa dizer ao computador a cor do ponto no comando RESET.

Digite:

```
150 RESET (38,10)
```

e execute o programa. O que você deve ver na tela agora é o desenho anterior, sem o olho direito. RESET diz ao computador para apagar o ponto situado na horizontal 38 e na vertical 10. Este ponto é o olho direito.

Para fazê-lo piscar, usamos os comandos SET e RESET no olho direito outra vez, acrescentando a linha 160:

```
160 GOTO 140
```

Verifique o seu programa para confirmar se ele ainda está igual ao nosso:

```
 5    CLS(0)

10    FOR H = 15 TO 48
20    SET(H,5,5)
30    SET(H,20,5)
40    NEXT H

50    FOR V = 5 TO 20
60    SET (15,V,5)
70    SET (48,V,5)
80    NEXT V ____________________________  rosto

90    SET (32,13,8) _____________________  nariz

100   FOR H = 28 TO 36
110   SET (H,16,4) ______________________  boca
120   NEXT H

130   SET (25,10,3)
140   SET (38,10,3) _____________________  olhos
```

```
150   RESET (38,10)  ________________  piscador
160   GOTO 140
```

e execute-o. Tente fazer um desenho à mão.

Estamos certo que você tem grandes habilidades artísticas.

O PONTO SALTADOR

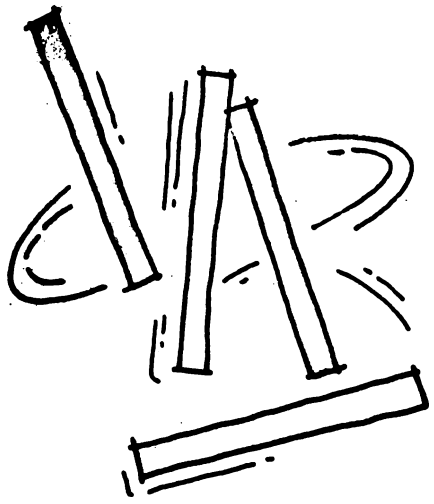
Usando SET e RESET podemos fazer desenhos animados. Digite e execute as linhas abaixo para fazer o ponto descer:

Lembre-se: ao mudar de programa, use NEW.

```
 5  CLS (0)
10  FOR V = 0 TO 31
20  SET (31,V,3)
30  RESET (31,V)
40  NEXT V
```

Todos os pontos mostrados pela linha 20 são apagados pela linha 30. Acrescente estas linhas para fazer o ponto subir:

```
50  FOR V = 31 TO 0 STEP -1
60  SET (31,V,3)
70  RESET (31,V)
80  NEXT V
```



A linha abaixo fará o ponto subir e descer várias vezes:

```
90 GOTO 10
```

Execute. Para retardar o movimento do ponto, mude as linhas 30 e 70:

```
30 IF V  >   0 THEN RESET (31,V - 1)
70 IF V  <  31 THEN RESET (31,V + 1)
```

o sinal > tem o mesmo significado da matemática, isto é, maior que. O sinal < significa menor que.

SET e RESET permitem fazer várias coisas, movimento de alvos, desenhos animados, etc. Use sua imaginação para explorá-los.

SE VOCÊ POSSUI JOYSTICKS...

Se possuir joysticks, existem outras opções para você. Se eles ainda não foram conectados, faça-o agora, ligando-os na parte de trás do computador. Não se preocupe, não é possível cometer erros porque eles só se encaixam nas aberturas corretas.

Agora digite este pequeno programa de demonstração:

```
10 CLS

20 PRINT  @   0,JOYSTK(0);
30 PRINT  @   5,JOYSTK(1);
40 PRINT  @  10,JOYSTK(2);
50 PRINT  @  15,JOYSTK(3);
60 GOTO 20
```

Não esqueça de digitar o ponto e vírgula no fim das linhas 20, 30, 40 e 50.

Execute o programa. Veja os quatro números na tela. Eles fornecem ao computador as coordenadas horizontais e verticais do manche dos dois joysticks.

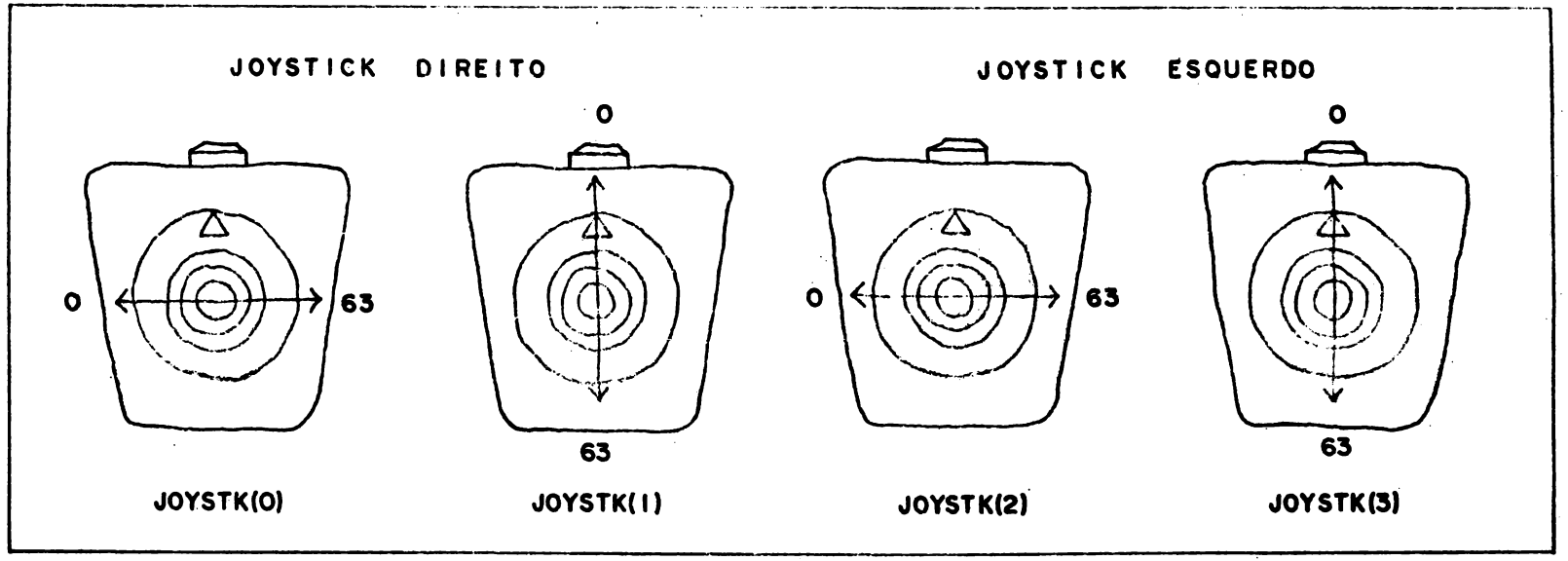
O segundo e o quarto número também podem mudar, mas não de 0 a 63.

Segure o manche do joystick direito, mantendo-o ao centro. Mova-o da esquerda para direita. O primeiro número da tela mudará, percorrendo os números 0 a 63.

Mova o manche do joystick esquerdo, da esquerda para a direita. Ele mudará o terceiro número na tela, também de 0 a 63.

Agora mova os manches de cima para baixo, mantendo-os no centro. Movendo o da direita de cima para baixo, você fará o segundo número mudar de 0 a 63. Movendo o da esquerda de cima para baixo, você fará o quarto número mudar, também, de 0 a 63.

Os desenhos a seguir, mostram como o computador lê a posição dos joysticks:



JOYSTK(0) e JOYSTK(1) lêem a posição do joystick direito:

- JOYSTK(0) lê a coordenada horizontal (da esquerda para direita).

- JOYSTK(1) lê a coordenada vertical (de cima para baixo).

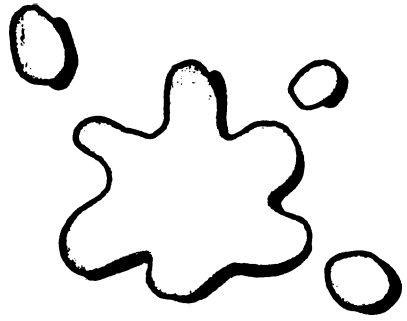
JOYSTK(2) e JOYSTK(3) lêem a posição do joystick esquerdo:

- JOYSTK(2) lê a coordenada horizontal.

- JOYSTK(3) lê a coordenada vertical.

Uma informação a mais. Elimine a linha 50 e execute o programa. Ele faz quase o mesmo , exceto por ler JOYSTK(3) - a posição vertical do joystick esquerdo.

Agora elimine a linha 20 e altere a 60 para:



```
60 GOTO 30
```

Execute o programa. Mova os manches para todos os lados e verá que nada acontece. O computador não lê nenhuma coordenada a não ser que você leia JOYSTK(0) primeiro.

Digite estas linhas:

```
20 A = JOYSTK(0)
60 GOTO 20
```

Execute o programa. Mesmo sem imprimir a posição de JOYSTK(0) o computador está lendo-a. Lembre-se que o computador pode ler as coordenadas de JOYSTK(1), JOYSTK(2) ou JOYSTK(3), mas antes deve ler JOYSTK(0).

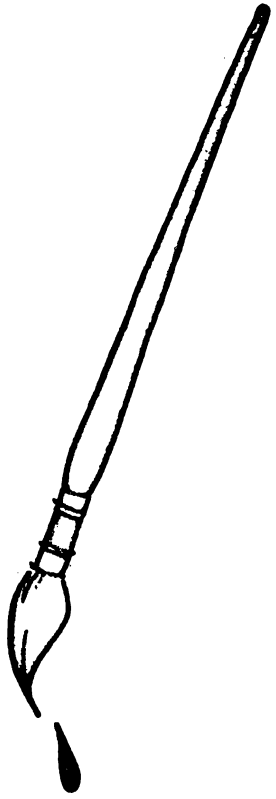
FAÇA RABISCOS COM OS JOYSTICKS

Digite:

```
10 CLS(0)
20 H = JOYSTK(0)
30 V = JOYSTK(1)
40 IF V > 31 THEN V = V - 32
80 SET (H,V,3)
90 GOTO 20
```

Execute.

> *> = significa maior ou igual a.*



Use o joystick direito para pintar um quadro. (Mova o manche devagar para que o computador tenha tempo de ler as coordenadas).

A linha 20 lê H, a posição horizontal do joystick direito, que pode ser um número de 0 a 63.

A linha 30 lê V, a posição vertical, do joystick direito, que também pode ser um número de 0 a 63. Como a posição vertical mais alta é 31, temos que acrescentar a linha 40 ao programa. A linha 40 faz V menor ou igual a 31.

A linha 80 coloca um ponto azul nas coordenadas H,V.

A linha 90 volta para pegar as próximas posições horizontal e vertical do joystick direito.

Ainda não usamos o joystick esquerdo. Vamos usá-lo, agora, para brincar com cores. Acrescente estas linhas:

```
50 C = JOYSTK(2)
60 IF C < 31 THEN C = 3
70 IF C > = 31 THEN C = 4
80 SET (H,V,C)
```

Execute o programa. Movendo o joystick esquerdo para a direita, o computador fará C = 4 e colocará pontos vermelhos na tela.

Movendo-o para esquerda, ele fará C = 3 e colocará pontos azuis.

Você quer usar os botões dos joysticks?

Acrescente estas linhas no fim do programa:

```
100 P = PEEK(65280)
110 PRINT P
120 GOTO 100
```

Agora digite:

RUN 100 (ENTER)

Isto diz ao computador para começar a execução do programa a partir da linha 100.

O computador deve estar imprimindo 255 ou 127 sem parar.

PEEK diz ao computador para ler o conteúdo de certa posição de memória, para ver que número ele contém. Ele leu o número que está na posição 65280. Como você não está pressionando nenhum dos botões, esta posição contém o número 255 ou 127.

Pressione o botão da direita. Agora esta posição de memória contém o número 126 ou 254.

Pressione o botão da esquerda. Agora esta posição de memória contém o número 125 ou 253.

Usando esta informação, você pode controlar o computador.

Faremos com que ele volte para a linha 10 (CLS(0) limpar a tela para preto) quando você pressionar o botão da direita.

Se você pressionar os botões quando não estiver executando o programa, surgirão, na tela, os caracteres @ ABCDEF ou HIJKLMN.

Altere as linhas 110 e 120:

```
110 IF P = 126 THEN 10
120 IF P = 254 THEN 10
```

Elimine a linha 90 e coloque esta:

```
130 GOTO 20
```

Execute o programa. Comece suas pinturas. Pressione o botão da direita para limpar a tela e começar outra vez.

ASSUNTOS APRENDIDOS NO CAPÍTULO 9

COMANDOS BASIC

SET

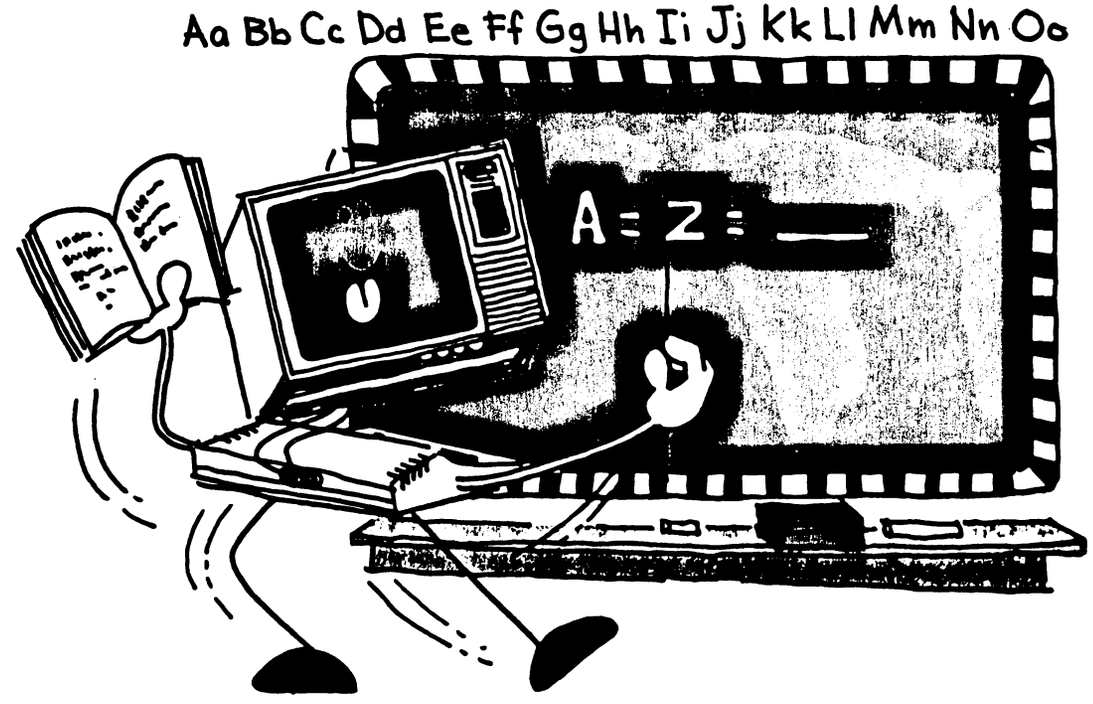
RESET

JOYSTK

PEEK

NOTAS

CAPÍTULO 10



UM PROFESSOR FANTÁSTICO

10

UM PROFESSOR FANTÁSTICO

O computador tem todos os atributos de quem já nasceu professor. Afinal de contas ele é paciente, incansável e consciente dos detalhes (... talvez um pouquinho brincalhão ...) . Dependendo do programador, (estamos falando de você) ele pode ser imaginativo, consolador ou até bastante entusiástico.

Então? Vamos adiante? Podemos usar RND para conseguir que o computador facilite nosso trabalho com a matemática. Digite:

```
10  CLS
20  X = RND(10)
30  Y = RND(10)
40  PRINT "QUANTO E' " X "*" Y
45  INPUT A
50  IF A = X *Y THEN 90
60  PRINT "A RESPOSTA E' " X*Y
70  PRINT "BOA SORTE DA PROXIMA VEZ"
```

```
80  GOTO 100
90  PRINT "CORRETO!!!"
100 PRINT "PRESSIONE <ENTER> QUANDO ESTIVER PRONTO PARA OUTRO TESTE"
105 INPUT A$
110 GOTO 10
```

Este programa checará seus conhecimentos na tabuada de multiplicação de 1 a 10.

Que alterações faria você, neste programa, para que o computador lhe ensinasse a somar números de 1 a 100?

FAÇA VOCÊ MESMO O SEU PROGRAMA

Aqui estão as linhas que mudamos:

```
20 X = RND(100)
30 Y = RND(100)
40 PRINT "QUANTO E' " X "+" Y
45 INPUT A
50 IF A = X + Y THEN 90
60 PRINT "A RESPOSTA E' " X + Y
```

Para tornar o programa um pouco mais interessante, podemos fazer com que o computador armazene a soma de todas as respostas corretas. Digite:

```
15 T = T + 1
95 C = C + 1
98 PRINT "EM" T "PERGUNTAS, VOCE ACERTOU" C
```

O T armazena o número de perguntas feitas pelo computador. Quando você executa o programa pela primeira vez, T é igual a 0 e toda vez que o computador passa pela linha 15, ele acrescenta 1 a T.

O C faz a mesma coisa. Ele armazena o número de respostas corretas e como encontra-se na linha 95, o computador só incrementa o seu valor quando você acertar uma resposta.

Existem várias maneiras de tornarmos este programa mais interessante. Acrescente novas linhas para o computador executar as seguintes tarefas:

Quando você liga o computador, todas as variáveis numéricas são inicializadas com 0 (zero), o mesmo ocorre quando você digita **NEW** (ENTER).

1 - Chame-o pelo nome.

2 - Anuncie uma resposta correta com um som e mostre uma cor.

3 - Imprima o problema e as mensagens formatadas na tela (use PRINT @ para isto).

4 - Armazene o percentual de respostas corretas.

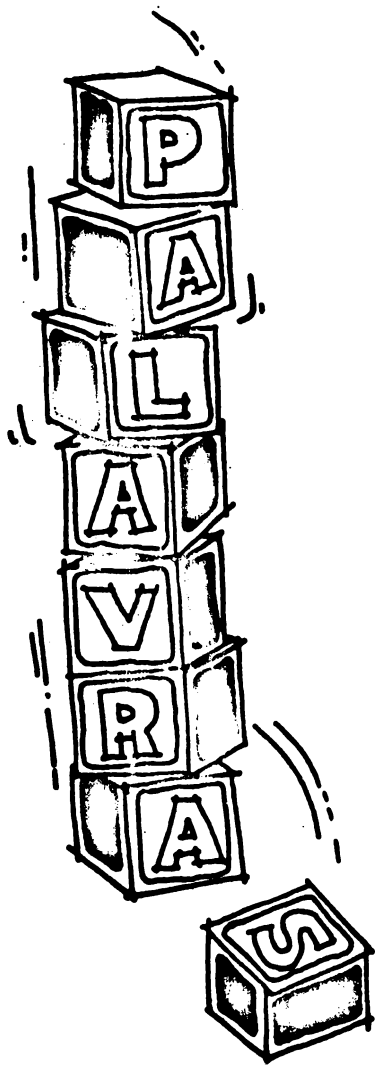
5 - Termine o programa se você conseguir 10 respostas corretas.

Você poderia criar outras variações deste programa. O computador poderia, por exemplo, fazer perguntas sobre negócios.

Use sua imaginação para isto. No fim deste manual encontra-se um programa que faz todas as tarefas acima. (apêndice I)

FAÇA VOCÊ MESMO O SEU PROGRAMA

PRIMEIRO CRIE O VOCABULÁRIO DO SEU COMPUTADOR



Para criar o vocabulário do seu computador (de modo que ele também possa contribuir para o seu), digite e execute este programa:

```
10 DATA UVAS, LARANJAS, PERAS
20 FOR X = 1 TO 3
30 READ F$
40 NEXT X
```

Então, o que aconteceu? Nada? Nada que você possa ver. Para ver o que o computador está fazendo, acrescente esta linha e execute:

```
35 PRINT "F$ = :" F$
```

A linha 30 diz ao computador para:

1 - Procurar uma linha com o comando DATA;

2 - Ler o primeiro item da lista, UVAS;

3 - Dar a UVAS o nome F$;

4 - Riscar UVAS da linha DATA;

Na segunda passagem pela linha 30 ele é orientado para fazer as mesmas coisas:

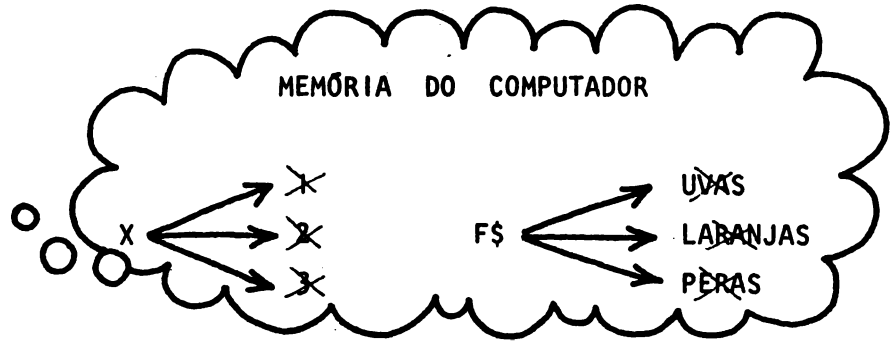
1 - Procurar uma linha com o comando DATA;

2 - Ler o primeiro item, agora é LARANJAS;

3 - Dar a LARANJAS o nome F$;

4 - Riscar LARANJAS da linha DATA;

Veja o que ocorre na memória do computador quando você executa o programa:



O que acontecerá se o computador executar a linha 30 outra vez?

Todos os ítens da linha DATA já estão riscados...

Digite:

```
60 GOTO 10
```

e execute o programa. Ele imprime ?OD ERROR IN 30. OD significado,sem dados (Out of Data). O computador já riscou todos os dados.

Lembra-se de como fazer o computador parar temporariamente, quando está executando um programa? Pressione (SHIFT) (©). Para continuar pressione qualquer tecla.

Digite esta linha e execute o programa:

```
50 RESTORE
```

Agora é como se o computador não tivesse riscando nada. A lista foi restaurada e o computador pode lê-la outra vez.

Uma característica interessante das linhas DATA é que elas podem ser colocadas em qualquer lugar do programa. Execute cada um destes programas:

```
10 DATA UVAS
20 DATA LARANJAS
30 FOR X = 1 TO 3
40 READ F$
50 PRINT "F$ = : " F$
60 NEXT X
70 DATA PERAS
```

```
10 DATA UVAS, LARANJAS
20 DATA PERAS
30 FOR X = 1 TO 3
40 READ F$
50 PRINT "F$ = : " F$
60 NEXT X
```

```
30 FOR X = 1 TO 3
40 READ F$
50 PRINT "F$ = : " F$
60 NEXT X
70 DATA UVAS
80 DATA LARANJAS
90 DATA PERAS
```

```
30 FOR X = 1 TO 3
40 READ F$
50 PRINT "F$ = : " F$
60 NEXT X
70 DATA UVAS, LARANJAS
80 DATA PERAS
```

Eles todos fazem a mesma coisa, não é? Esta característica deve ser útil para alguma coisa, você não concorda conosco?

...AGORA, MANDE-O FAZER SEU VOCABULÁRIO

"Ótimo"

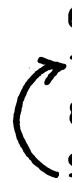
Aqui estão algumas palavras e definições sobre as quais você gostaria de ser testado, temos certeza:

PALAVRAS	DEFINIÇÕES

```
10 DATA TACITURNO, QUEM HABITUALMENTE NAO FALA
20 DATA LOQUAZ, QUEM FALA MUITO
30 DATA VOCIFERANTE, QUEM FALA ALTO E VEEMENTEMENTE
40 DATA BREVE, CONCISO
50 DATA EFUSIVO, DEMONSTRATIVO OU EXTROVERTIDO
```

Agora, o computador deverá escolher uma palavra da lista.

Bem, há dez ítens na lista. Talvez isto funcione, vamos tentar:



```
60  N = RND(10)
70  FOR X = 1 TO N
80  READ A$
90  NEXT X
100 PRINT "A PALAVRA ESCOLHIDA E':" A$
```

Execute o programa várias vezes para ver se funciona.

Não funcionou como queríamos. O computador pode escolher tanto uma palavra quanto uma definição. O que realmente queremos é que ele pegue uma palavra dos ítens: 1, 3, 5, 7 e 9.

Felizmente existe uma forma de indicarmos isto ao computador. Digite:

```
65 IF INT (N/2) = N/2 THEN N = N - 1
```

Execute o programa algumas vezes e verá que deve funcionar. INT diz ao computador para considerar somente a parte inteira do número e ignorar a parte decimal. Por exemplo, o computador interpreta INT (3.9) como 3.

Eis como funciona a linha 65. Digamos que o computador pegue o número randômico 10. Ele faz este cálculo:

```
INT (10/2) = 10/2
INT (5) = 5
   5 = 5
```

Como isto é verdade, 5 é igual a 5, o computador executa o THEN da linha e faz N igual à 9 (10-1).

Contudo, se o computador pegar o 9, ele fará:

```
INT (9/2) = 9/2
INT (4.5) = 4.5
     4 = 4.5
```

Como não é verdade, pois 4 não é igual a 4.5, o computador não subtrai 1 de N e o valor continua igual a 9.

Agora que o computador sabe pegar aleatoriamente uma palavra, ele também deve saber ler sua definição. Basta, para isto, acrescentar estas linhas no fim do programa:

```
110 READ B$
120 PRINT "A DEFINICAO E':" B$
```

Agora, execute o programa várias vezes. Acrescente esta linha no começo do programa:

```
5  CLEAR  100
```

para dar ao computador espaço suficiente para armazenar as palavras. Coloque estas linhas no fim do programa:

```
130 RESTORE
140 GOTO 60
```

para permitir que o computador pegue uma nova palavra e sua definição,na lista completa de ítens.

Eis como ficou o novo programa:

```
5   CLEAR 100

10  DATA TACITURNO, QUEM HABITUALMENTE NAO FALA
20  DATA LOQUAZ, QUEM FALA MUITO
30  DATA VOCIFERANTE, QUEM FALA ALTO E VEEMENTEMENTE
40  DATA BREVE, CONCISO
50  DATA EFUSIVO, DEMONSTRATIVO OU EXTROVERTIDO
60  N = RND(10)
65  IF INT (N/2) = N/2 THEN N = N-1
70  FOR X = 1 TO N
80  READ A$
90  NEXT X
100 PRINT "A PALAVRA ESCOLHIDA E':" A$
110 READ B$
120 PRINT "SUA DEFINICAO E': " B$
130 RESTORE
140 GOTO 60
```

Se você quiser, acrescente outras palavras e definições, basta colocar mais linhas de DATA.

Para variações deste programa, use: Estados e Capitais, Cidades e Países, palavras estrangeiras e suas traduções. Sugira outras idéias.

Quer completar este programa? Tente antes de virar a próxima página, pois o nosso está lá.
O programa executa as seguintes tarefas:

1 - Imprime somente a definição;

2 - Pergunta qual é a palavra;

3 - Compara a sua resposta com a palavra correspondente à definição escolhida;

4 - Diz se sua resposta está correta. Se estiver errada, imprime a correta.

FAÇA VOCÊ MESMO O SEU PROGRAMA

Aqui está o nosso:

```
5   CLEAR 500
10  DATA TACITURNO, QUEM HABITUALMENTE NAO FALA
20  DATA LOQUAZ, QUEM FALA MUITO
30  DATA VOCIFERANTE, QUEM FALA ALTO E VEEMENTEMENTE
40  DATA BREVE, CONCISO
50  DATA EFUSIVO, DEMONSTRATIVO OU EXTROVERTIDO
60  N = RND(10)
65  IF INT (N/2) = N/2 THEN N = N-1
70  FOR X = 1 TO N
80  READ A$
90  NEXT X
110 READ B$
120 PRINT "QUAL E' A PALAVRA QUE SIGNIFICA: " B$
130 RESTORE
140 INPUT R$
150 IF R$ = A$ THEN 190
160 PRINT "ERRADO"
170 PRINT "A PALAVRA CORRETA E':" A$
180 GOTO 60
190 PRINT "CORRETO"
200 GOTO 60
```

Sinta-se à vontade para melhorar a apresentação do programa, isto é, formato de tela, som, etc...

ASSUNTOS APRENDIDOS NO CAPÍTULO 10

COMANDOS BASIC

DATA

READ

RESTORE

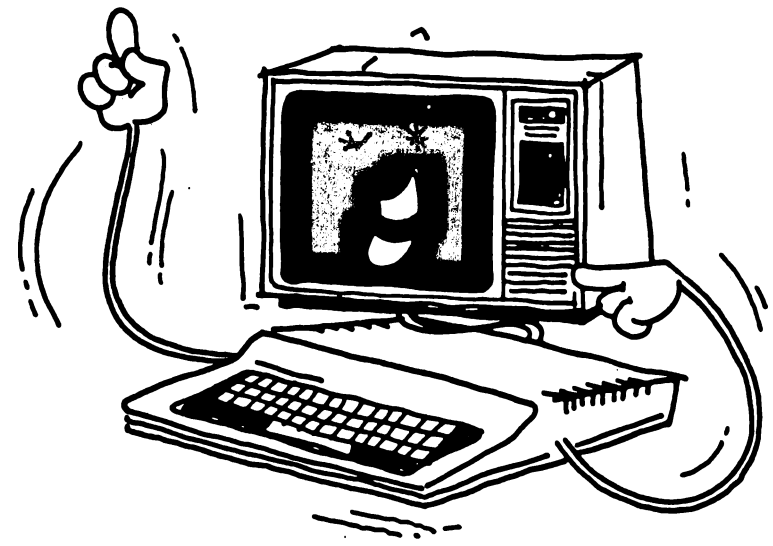
INT

CLEAR

NOTAS

CAPÍTULO II

$$Ax\,(BY + C) - D + E\,(G/W) - F$$



AJUDA COM A MATEMÁTICA

11

AJUDA COM A MATEMÁTICA

Resolver problemas matemáticos complexos com rapidez e precisão, é uma área em que o computador "dá show". Mas antes de você ficar louco digitando grandes fórmulas matemáticas , leia estas dicas. Elas lhe serão muito úteis.

Um método simples para manipular fórmulas complexas é usar SUBROTINAS. Digite este programa:

```
10  PRINT "EXECUTANDO O PROGRAMA PRINCIPAL"
20  GOSUB 500
30  PRINT "DE VOLTA AO PROGRAMA PRINCIPAL"
40  END

500 PRINT "EXECUTANDO A SUBROTINA"
510 RETURN
```

A linha 20 diz ao computador para ir para a subrotina na linha 500. RETURN diz ao computa

dor para voltar para o comando BASIC, imediatamente após o GOSUB.

Apague a linha 40 para ver o que acontece quando executar o programa.

Apagou?

Se você o fez, a tela deve conter estas mensagens:

```
EXECUTANDO O PROGRAMA PRINCIPAL
EXECUTANDO A SUBROTINA
AGORA DE VOLTA AO PROGRAMA PRINCIPAL
EXECUTANDO A SUBROTINA
?RG ERROR IN 510
```

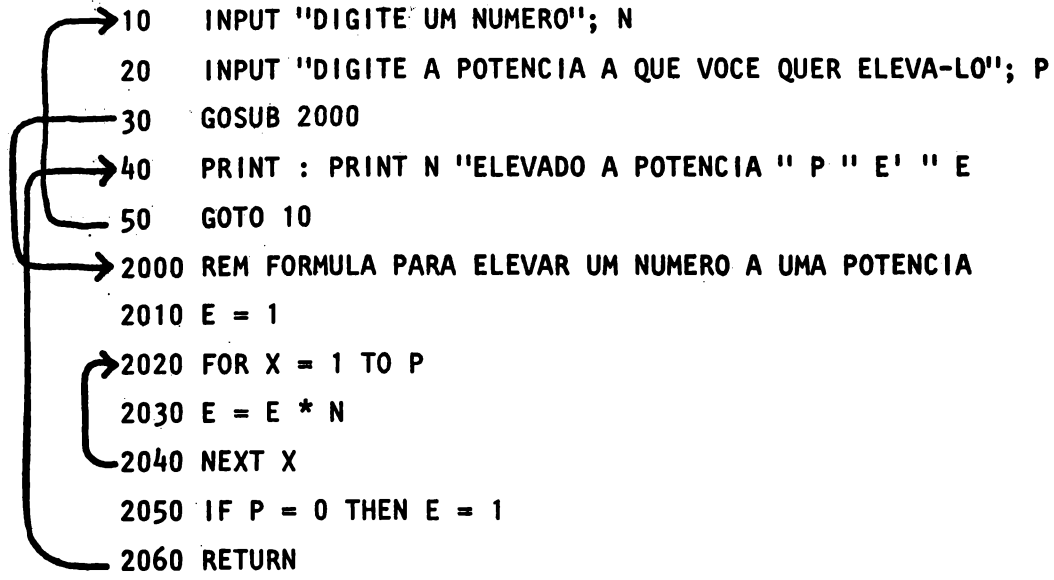
RG significa RETURN sem GOSUB. Você descobriu por que a falta da linha 40 (END)causou erro?

Primeiro, o computador foi mandado para a subrotina na linha 500 pelo GOSUB e depois voltou para o comando BASIC imediatamente abaixo do GOSUB.

Então, como você apagou o END, ele foi para a próxima linha do programa que era a subrotina. Quando alcançou o RETURN, ele ficou sem saber para onde voltar, pois desta vez ele não tinha sido enviado para a subrotina pelo GOSUB.

Aqui está uma subrotina que eleva um número à potência que você desejar:

Você está vendo alguma coisa diferente no INPUT ? Podemos mandar o computador imprimir uma mensagem antes de esperar pela informação a ser digitada.



```
10   INPUT "DIGITE UM NUMERO"; N
20   INPUT "DIGITE A POTENCIA A QUE VOCE QUER ELEVA-LO"; P
30   GOSUB 2000
40   PRINT : PRINT N "ELEVADO A POTENCIA " P " E' " E
50   GOTO 10
2000 REM FORMULA PARA ELEVAR UM NUMERO A UMA POTENCIA
2010 E = 1
2020 FOR X = 1 TO P
2030 E = E * N
2040 NEXT X
2050 IF P = 0 THEN E = 1
2060 RETURN
```

Repare que introduzimos duas coisas novas no programa.

Veja a linha 40; se achar mais fácil, você pode combinar duas ou mais linhas dentro de uma só, usando dois pontos para separá-las. A linha 40 contém as duas linhas:

PRINT, sozinho, diz ao computador para saltar uma linha.

```
PRINT
```

e

```
PRINT N "ELEVADO A POTENCIA "P" E' " E
```

A linha 2000 também tem alguma coisa nova: o REM.

REM não significa nada para o computador, ele ignora qualquer linha que comece com REM . Você pode colocá-la em qualquer lugar no programa, como comentário para lembrar-se do que o programa faz. Estas linhas REM não fazem diferença para a execução do programa; são tratadas pelo computador como simples comentários.

Se você não acredita, acrescente estas linhas e execute o programa:

```
5   REM ESTE E' UM PROGRAMA PECULIAR
17  REM ESTA LINHA DEVE BAGUNCAR O PROGRAMA
45  REM A PROXIMA LINHA MANTEM O SUBPROGRAMA SEPARADO
```

Satisfeito? ... Muito bom!!!

Mude o programa de modo que o computador imprima uma tabela de quadrados (um número elevado a 2) para números de 2 a 10.

FAÇA VOCÊ MESMO O SEU PROGRAMA

Nossa resposta está no apêndice I no fim deste manual.

DÊ UMA AJUDA AO COMPUTADOR

À medida que as fórmulas matemáticas ficam mais complexas, o computador precisa de uma ajuda para entendê-las. Por exemplo, o que acontece se você pedir ao computador para resolver o problema a seguir?

Divida 13+ 3 por 8.

A palavra operação significa alguma coisa que você está mandando o computador fazer. Aqui estamos falando de operações de adição, subtração, multiplicação e divisão.

Você espera que ele faça o seguinte:

13+3/8 = 16/8 = 2

Mas ele fará diferente. Digite esta linha:

PRINT 13+3/8 (ENTER)

O que o computador fez foi lógico, mas de acordo com suas próprias regras.

REGRAS MATEMÁTICAS

O computador resolve as fórmulas matemáticas nesta ordem:

1 - Qualquer operação de multiplicação e divisão é resolvida primeiro.
2 - Operações de adição e subtração são resolvidas por último.
3 - No caso de existir mais de uma multiplicação/divisão ou adição/subtração, as operações são executadas da esquerda para a direita.

No problema anterior, o computador seguiu suas regras. Ele executou a divisão primeiro (3/8 = 0.375) e depois a adição (13+0.375 = 13.375). Para que ele faça de maneira diferente, podemos usar parênteses. Digite esta linha:

PRINT (13 + 3) / 8 (ENTER)

Sempre que o computador vê uma operação entre parênteses, ele a resolve antes de qualquer outra.

O que você acha que o computador imprimirá como resposta para estes problemas?

EXERCÍCIOS

PRINT 10 - (5 - 1) / 2 _____

PRINT 10 - 5 - 1 / 2 _____

PRINT (10 - 5 - 1) / 2 _____

PRINT (10 - 5) - 1 / 2 _____

PRINT 10 - (5 - 1 / 2) _____

Acabou? Digite cada uma das linhas para verificar suas respostas.

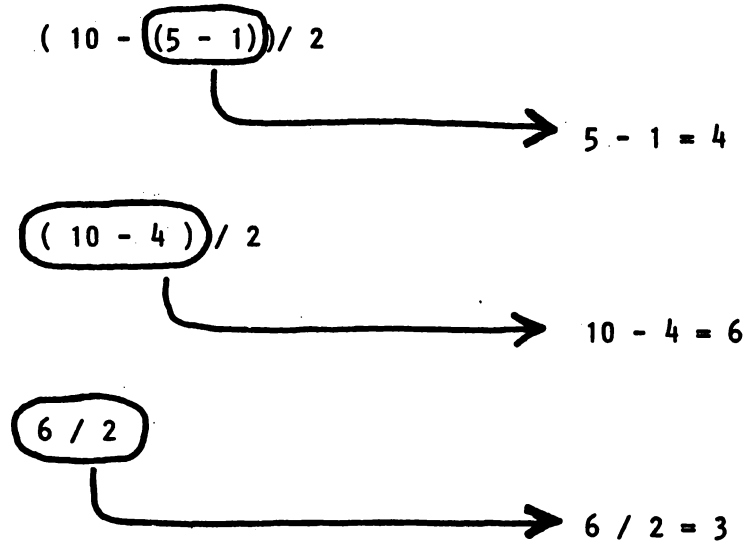
O que acontecerá se você pedir ao computador para resolver a expressão:

Divida, 10 menos a diferença de 5 menos 1, por 2.

Isto é o que você realmente está pedindo para ele fazer:

(10 - (5 - 1)) / 2

Quando o computador encontra um problema com mais de um parênteses, ele resolve primeiro o mais interno e depois o mais externo. Em outras palavras, faz o seguinte:



REGRAS PARA PARÊNTESES

1 - Quando o computador encontra uma expressão contendo parênteses, ele resolve primeiro as operações dentro dos parênteses e depois as outras.

2 - Se houver parênteses dentro de parênteses, o computador resolve primeiro os mais internos e depois o resto.

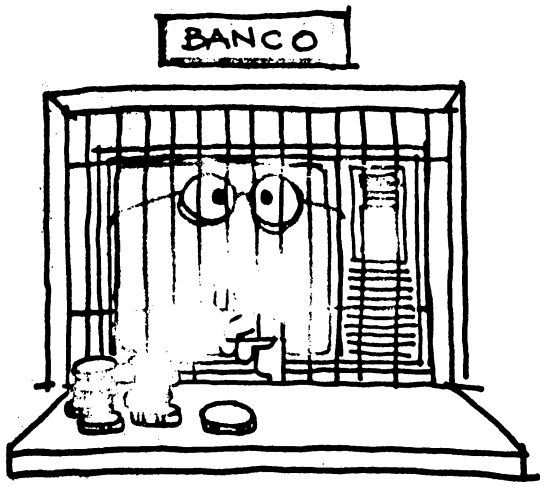
Coloque parênteses no problema abaixo para que o computador imprima 28 como resposta.

EXERCÍCIO

```
PRINT 30 - 9 - 8 - 7 - 6
```

Resposta: PRINT 30 - (9 - (8 - (7 - 6)))

Com o que foi ensinado neste capítulo, você pode pedir ao computador que faça todos os seus problemas matemáticos. O programa a seguir usa duas subrotinas, e é útil para aqueles que economizam depositando no banco a mesma quantia de dinheiro, todos os meses.



"Economizei dez centavos"

```
10    INPUT "SEU DEPOSITO MENSAL E' "; D
20    INPUT "O JURO BANCARIO ANUAL E' "; I
30    I = I/12 *.01
40    INPUT "NUMERO DE DEPOSITOS"; P
50    GOSUB 1000
60    PRINT "VOCE TERA' CR$" FV "EM" P "MESES"
70    END

1000  REM FORMULA MENSAL DE JUROS
1010  N = 1 + I
1020  GOSUB 2000
1030  FV = D * ((E - 1)/I)
1040  RETURN

2000  REM FORMULA PARA ELEVAR UM NUMERO A UMA POTENCIA
2010  E = 1
2020  FOR X = 1 TO P
2030  E = E * N
2040  NEXT X
2050  IF P = 0 THEN E = 1
2060  RETURN
```

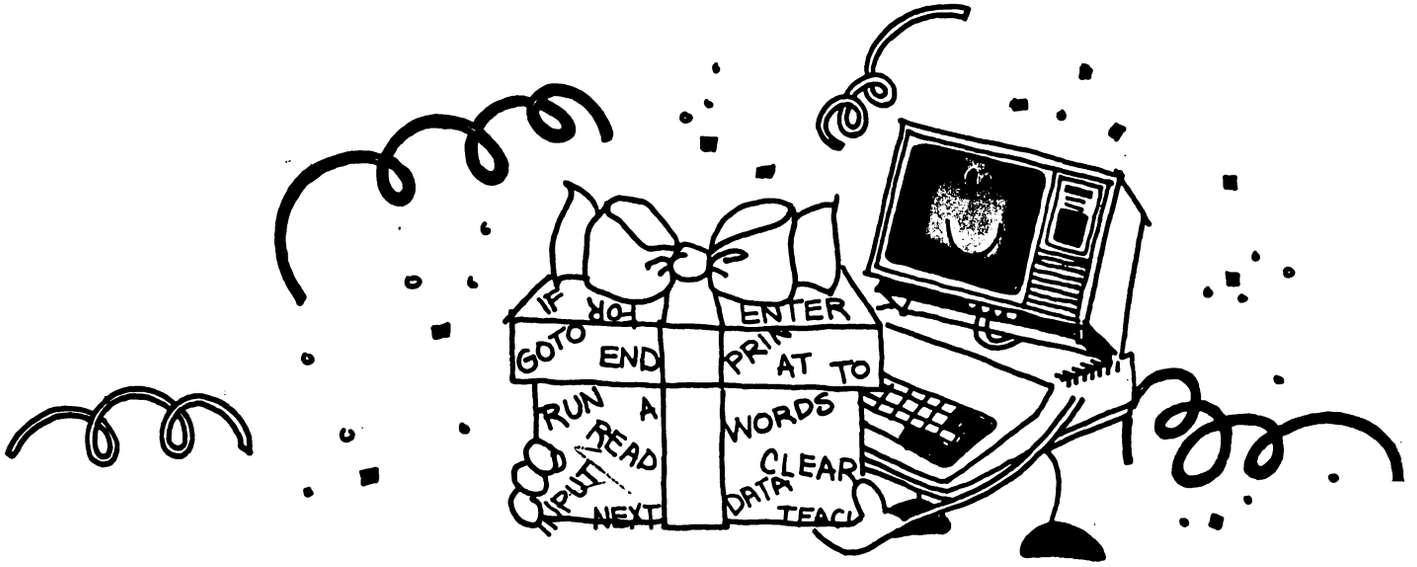
Veja que temos uma subrotina chamando outra subrotina. Isto é perfeitamente possível para o computador, desde que tenhamos um GOSUB para levar o computador para cada subrotina e um RETURN em cada subrotina para retornar ao comando BASIC logo abaixo de cada GOSUB.

ASSUNTOS APRENDIDOS NO CAPÍTULO 11

COMANDOS BASIC	SÍMBOLOS BASIC	CONCEITOS BASIC
GOSUB	( )	Ordem das operações matemáticas.
RETURN	:	Regras de parênteses.
REM		

NOTAS

CAPÍTULO 12



HABILIDADE COM PALAVRAS

12

HABILIDADE COM PALAVRAS

Você não está impressionado? Bem, mais tarde mostraremos um modo prático para usar tal habilidade.

Uma das maiores habilidades do computador é sua facilidade de trabalhar com palavras, ele pode sem se cansar, combinar ou separar palavras da forma que você quiser. Graças a este dom, ele pode ser ensinado a ler, escrever ou manter uma conversação.

Para os iniciantes, vejam o que acham disto:

```
10  PRINT "DIGITE UMA SENTENCA"
20  INPUT S$
30  PRINT "SUA SENTENCA TEM" LEN(S$) "CARACTERES"
40  INPUT "QUER TENTAR OUTRA?" ; A$
50  IF A$ = "SIM" THEN 10
```

Impressionado? LEN(S$) diz ao computador para contar o tamanho do string S$ —— a sentença. Ele obedientemente conta todos os caracteres da sentença, incluindo espaços e pontuações.

Aqui vai outra habilidade. Apague tudo e digite este programa para ele compor um poema.

```
10  A$ = "UMA ROSA"
20  B$ = "  "
30  C$ = "E' UMA ROSA"
40  D$ = B$ + C$
50  E$ = "E ASSIM POR DIANTE"
60  F$ = A$ + D$ + B$ + E$
70  PRINT F$
```

Agora, o computador combinou strings. O sinal "mais" diz para fazer isto. D$ combina B$ e C$ para formar " E' UMA ROSA " e você pode ver quais strings são combinados para formar F$.

Um alerta sobre combinação de strings. Acrescente esta linha e execute:

```
80  G$ = F$ + F$ + F$ + F$ + F$ + F$ + F$
```

Quando executar este programa, o computador imprime: ?OS ERROR IN 80. O OS significa que acabou o espaço para armazenar strings. O computador reserva somente 200 caracteres para trabalhar com strings. Acrescente esta linha no começo do programa para conseguir mais espaço para strings:

```
5   CLEAR  500
```

Execute o programa outra vez. Isto resolve o primeiro problema, mas ainda há outro.

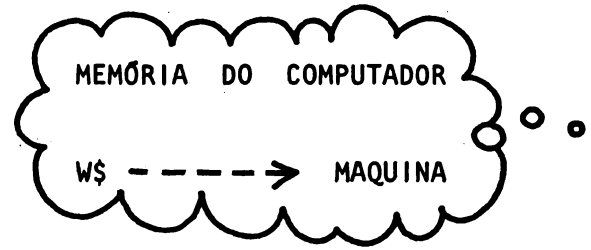
Desta vez o computador imprime: ?LS ERROR IN 80. LS significa string muito longo. A linha 80 pede ao computador para formar um string - G$ - com mais de 255 caracteres. Ele não pode se lembrar de strings com tantos caracteres.

Agora que o computador combinou strings, vamos pedir que os separe. Digite e execute este programa:

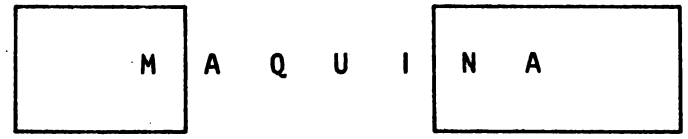
```
10  INPUT "DIGITE UMA PALAVRA"; W$
20  PRINT "A PRIMEIRA LETRA E':" LEFT$(W$,1)
30  PRINT "AS DUAS ULTIMAS LETRAS SAO:" RIGHT$(W$,2)
40  GOTO 10
```

Eis o que o computador está fazendo:

Na linha 10 você fornece um string para W$. Vamos supor que o string seja "MAQUINA".



O computador então desenvolve vários cálculos nas linhas 20 e 30 para conseguir a primeira letra à esquerda e as duas últimas letras do string .



LEFT$ (W$, 1) RIGHT$ (W$, 2)

Execute o programa até se cansar.

Acrescente esta linha:

```
5  CLEAR 500
```

para que o computador reserve um espaço maior para trabalhar com strings. Agora introduza uma sentença ao invés de uma palavra.

Que mudança faria você nas linhas 20 e 30 para que o computador forneça as cinco primeiras letras e as seis últimas da sentença?

EXERCÍCIO DE PROGRAMAÇÃO

20 _____

30 _____

Respostas:

```
20 PRINT "AS CINCO PRIMEIRAS LETRAS SAO: " LEFT$(W$,5)
30 PRINT "AS SEIS ULTIMAS LETRAS SAO: " RIGHT$(W$,6)
```

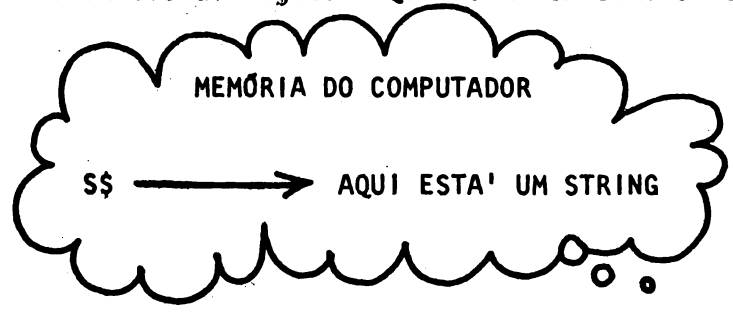
Apague o programa e digite este:

Você ainda se lembra como se apaga um programa? Digite: NEW (ENTER).

```
10 CLEAR 500
20 INPUT "DIGITE UMA SENTENCA"; S$
30 PRINT "DIGITE UM NUMERO ENTRE 1 E" LEN (S$)
40 INPUT X
50 PRINT "O TRECHO  COMECARA' NA POSICAO" X
60 PRINT "DIGITE OUTRO NUMERO ENTRE 1 E" LEN(S$) - X + 1
70 INPUT Y
80 PRINT "O TRECHO TERA' " Y "CARACTERES"
90 PRINT "ESTE MIDSTRING E':" MID$ (S$,X,Y)
100 GOTO 20
```

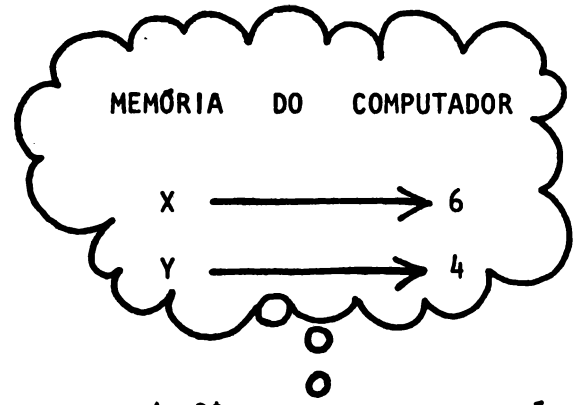
Execute este programa várias vezes para entender como funciona MID$;

Eis como o programa funciona. Digite "AQUI ESTA' UM STRING" como sua sentença.

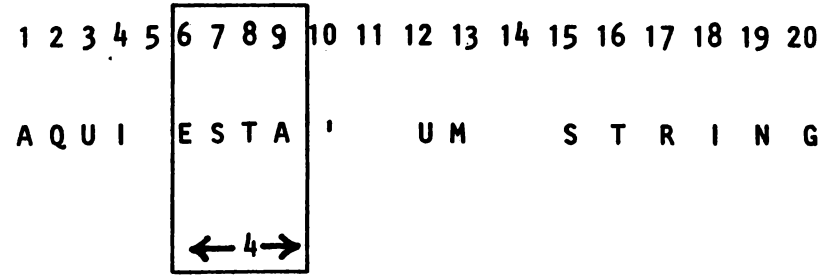


Na linha 30 o computador calcula primeiro o comprimento de S$ ____ 20 caracteres. Então e le manda escolher um número entre 1 e 20. Suponhamos que você escolha o número 6.

O computador, na linha 60, pede para fornecer um outro número entre 1 e 15 (20 - 6 + 1). Suponhamos que escolha o número 4.



Na linha 90 ele imprime a parte de S$, que começa no caráter 6 e tem um tamanho de 4 ca - racteres.



MID$ (S$, 6, 4)

A seguir apresentamos outra coisa que você pode fazer com MID$.

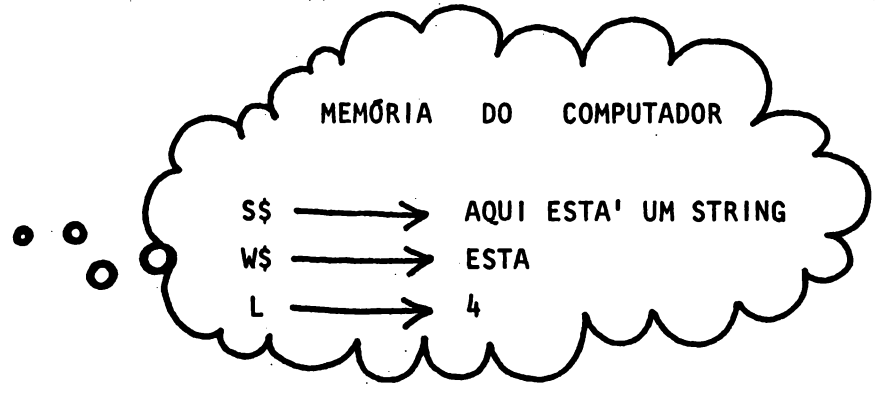
Apague o programa e digite:

Estes tipos de programas podem ser usados para grandes arquivos de informação. Por exemplo, pesquisando strings, você poderia selecionar uma lista de endereços de clientes de São Paulo.

```
10  INPUT "DIGITE UMA SENTENCA"; S$
20  INPUT "DIGITE UMA DAS PALAVRAS DA SENTENCA"; W$
30  L = LEN(W$)
40  FOR X = 1 TO LEN(S$)
50  IF MID$ (S$,X,L) = W$ THEN 90
60  NEXT X
70  PRINT "SUA PALAVRA NAO CONSTA DA SENTENCA"
80  END
90  PRINT W$ "COMECA NO CARATER NUMERO" X
```

Execute este programa várias vezes. Eis como funciona:

Suponhamos que você tenha digitado, a sentença anterior e a palavra que você definiu para W$ é "ESTA". Na linha 30 o computador então calcula o tamanho de W$ - 4 caracteres.



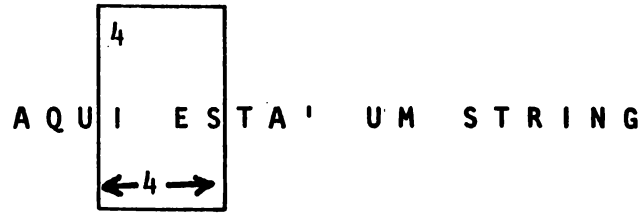
Ele então vai, através do loop FOR/NEXT (linhas 40 a 90), contando cada caráter em S$, começando com o caráter 1 e terminando com o caráter número LEN(S$) - 20.

Toda vez que conta um novo caráter, o computador determina uma nova parte de S$. Cada parte começa no caráter X (1 a 20) e possui L (4) posições de comprimento.

Por exemplo, quando X é igual a 1, o computador determina esta parte:

MID$ (S$,1,4)

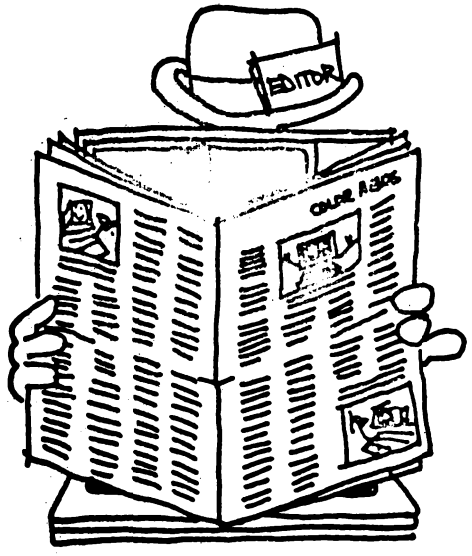
No quarto loop, quando X é igual a 4, o computador determina esta parte:



MID$ (S$,4,4)

Ele finalmente acha a parte que está procurando, quando X é igual a 6.

ELE PODE SER UM EDITOR



"#@%!&$!"

Você pode programar o computador para ajudá-lo a melhorar sua escrita e economizar horas de datilografia e revisões.

Suponhamos que você queira modificar uma frase:

```
10  A$ = "MODIFICAR UMA SENTENCA"
```

Complete este programa de modo que o computador coloque estas palavras no começo da sen - tença:

```
"E' FACIL"
```

e imprima a nova sentença:

```
E' FACIL MODIFICAR UMA SENTENCA.
```

FAÇA VOCÊ MESMO O SEU PROGRAMA

Aqui está o nosso programa:

```
10 A$ = "MODIFICAR UMA SENTENCA"
20 B$ = "E' FACIL"
30 C$ = B$ + " " + A$
40 PRINT C$
```

Agora verifique se consegue modificar o programa para que o computador:

1 - Descubra a posição inicial do string: UMA SENTENCA

2 - Apague as palavras UMA SENTENCA, formando um novo string: E' FACIL MODIFICAR

3 - Acrescente estas palavras no fim do novo string: QUALQUER COISA QUE VOCE QUEIRA

4 - Imprima o string recém formado: E' FACIL MODIFICAR QUALQUER COISA QUE VOCE QUEIRA

FAÇA VOCÊ MESMO O SEU PROGRAMA

OBS.: Para formar o string E' FACIL MODIFICAR, você precisa apanhar a porção à esquerda do string E' FACIL MODIFICAR UMA SENTENCA.

Resposta:

Este programa é a base para um programa de processamento de texto.

```
10  A$ = "MODIFICAR UMA SENTENCA"
20  B$ = "E' FACIL"
30  C$ = B$ + "  " + A$
40  PRINT C$
50  Y = LEN ("UMA SENTENCA")
60  FOR X = 1 TO LEN(C$)
70  IF MID$ (C$,X,Y) = "UMA SENTENCA" THEN 90
80  NEXT X
85  END
90  D$ = LEFT$ (C$, X - 1)
100 E$ = D$ + "QUALQUER COISA QUE VOCE QUEIRA"
110 PRINT E$
```

SE VOCÊ GOSTA DE DESAFIOS, TENTE ISTO...

Escreva um programa no qual:

1 - O computador peça para definir:

a) Uma sentença

b) Uma frase dentro da sentença para ser eliminada

c) Uma nova frase para substituir a sentença eliminada.

2 - O computador então imprima a nova sentença.

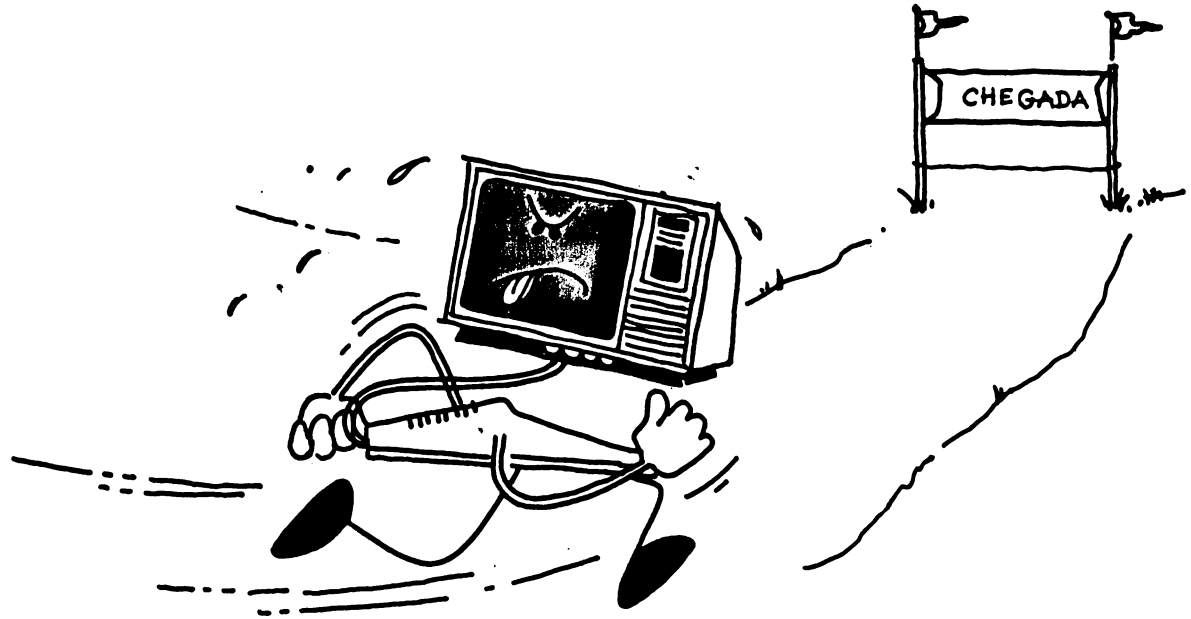
Isto pode levar tempo, mas você tem todos os conhecimentos necessários para escrevê-lo. Nossa resposta está no apêndice I no fim deste manual.

FAÇA VOCÊ MESMO O SEU PROGRAMA

ASSUNTOS APRENDIDOS NO CAPÍTULO 12

COMANDOS BASIC	OPERADOR DE STRING
LEN	+
LEFT$	
RIGHT$	
MID$	

CAPÍTULO 13



VENÇA O COMPUTADOR

13

VENÇA O COMPUTADOR

Você achará o computador mais útil se permitir que ele vigie e reaja a tudo que faz. Por "vigiar", queremos dizer vigiar o teclado para ver se você está pressionando alguma coisa. O comando INKEY$ permite isto.

Digite:

Você se lembra do significado de < >? Estes símbolos significam diferente de.

"" significa um string vazio.

```
10  A$ = INKEY$
20  IF A$< > "" GOTO 50
30  PRINT "VOCE NAO PRESSIONOU NADA"
40  GOTO 10
50  PRINT "VOCE PRESSIONOU A TECLA" A$
```

Pressione uma tecla durante a execução do programa.

INKEY$ faz o computador vigiar o teclado para ver se você pressionou alguma tecla. O computador faz isto com uma super velocidade. Para ter uma idéia da velocidade com que isto

é feito, basta saber que você não terá tempo de fazer nada, durante os vinte primeiros testes do computador.

O computador chama esta pseuda tecla de A$ e toma suas decisões baseado no seu conteúdo.

Se A$ = "" o computador imprime "VOCE NAO PRESSIONOU NADA" e volta à linha 10 para checar o teclado novamente. Contudo, se A$ for igual a alguma coisa, o computador vai à linha 50 e imprime a tecla que foi pressionada.

Para uma pesquisa contínua, digite a linha abaixo e execute o programa:

```
60 GOTO 10
```

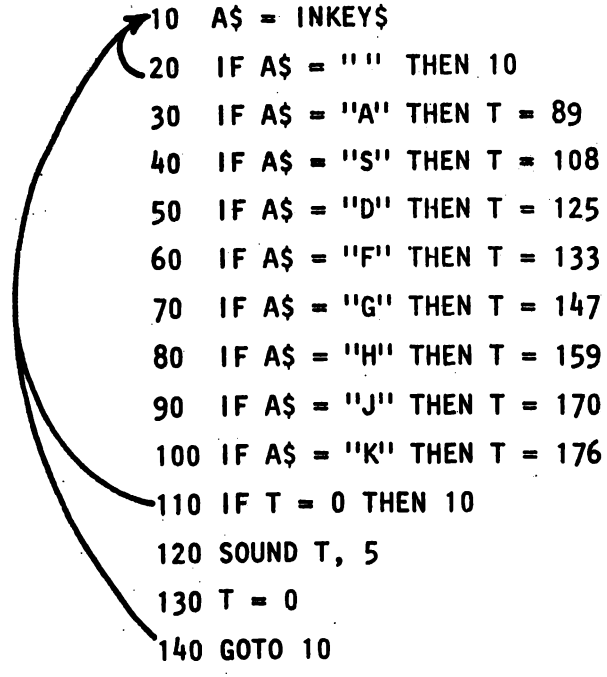
Não importa a sua velocidade, pois o computador é muito mais rápido. Para ver as teclas pressionadas, apague a linha 30.

UM PIANO ELETRÔNICO

Tente usar INKEY$ para transformar seu teclado num piano. Veja a tabela no apêndice A - tons musicais - . Ela mostra os números correspondentes aos tons do C médio até o C da próxima oitava.

```
C - 89  E - 125  G - 147  B - 170
D - 108 F - 133  A - 159  C - 176
```

Podemos pedir ao computador que, se você pressionar uma certa tecla, ele soe um dos tons. Apague o programa e digite:



```
10  A$ = INKEY$
20  IF A$ = "" THEN 10
30  IF A$ = "A" THEN T = 89
40  IF A$ = "S" THEN T = 108
50  IF A$ = "D" THEN T = 125
60  IF A$ = "F" THEN T = 133
70  IF A$ = "G" THEN T = 147
80  IF A$ = "H" THEN T = 159
90  IF A$ = "J" THEN T = 170
100 IF A$ = "K" THEN T = 176
110 IF T = 0 THEN 10
120 SOUND T, 5
130 T = 0
140 GOTO 10
```

Execute o programa... Bem, o que está esperando? Toque uma canção. Pressione uma das teclas localizadas na terceira fila do teclado ___ de A a K.

Como seria modificado o programa por este comando?

120 SOUND T,1

Por que este programa não funciona direito se fosse usado INPUT ao invés de INKEY$?

Usando INPUT o computador espera que você pressione (ENTER) para interpretar o que foi digitado. Com INKEY$ ele analisa a digitação imediatamente.

Existe uma outra forma de escrever este programa usando linhas READ e DATA. Sabe como isto pode ser feito?

Vai ficar assim:

```
10  A$ = INKEY$
20  FOR X = 1 TO 8
30  READ B$, T
40  IF A$ = B$ THEN SOUND T,5
50  NEXT X
60  RESTORE
70  GOTO 10
80  DATA A,89,S,108
90  DATA D,125,F,133
100 DATA G,147,H,159
110 DATA J,170,K,176
```

Usando DATA e READ fica mais fácil acrescentar mais tons ao repertório do computador

VENÇA O COMPUTADOR

Digite este programa:

```
10  X = RND(4)
20  Y = RND(4)
```

```
30  PRINT "QUANTO E' "X" + "Y
40  T = 0
50  A$ = INKEY$
60  T = T + 1
70  SOUND 128, 1
80  IF T = 15 THEN 200
90  IF A$ = "" THEN 50
100 GOTO 10

200 CLS(7)
210 SOUND 180,30
220 PRINT "MUITO TARDE"
```

Aqui está o que o programa faz:

As linhas 10 e 20 fazem o computador pegar dois números randômicos entre 1 e 4.

A linha 30 pergunta qual é a soma destes números.

A linha 40 inicializa T com 0. Usaremos T como um contador.

A linha 50 dá a primeira chance de responder à pergunta.

A linha 60 soma 1 ao contador. Na próxima passagem pela linha 60, ele adicionará, novamente, 1 ao contador para fazer T igual a 2. Cada vez que o computador executa a linha 60 ele adiciona 1 a T.

A linha 70 está lá para deixá-lo nervoso.

A linha 80 diz ao computador que você tem 15 chances para responder. Quando T for igual a 15, o tempo acabou e o computador informará isto nas linhas 200, 210 e 220.

A linha 90 diz ao computador para voltar à linha 50 se você não responder nada, dando-lhe assim outra oportunidade.

O computador vai à linha 100 se você responder; neste caso ele cria um outro problema.

Como faria você para obter três vezes mais tempo para responder as perguntas?

Resposta:

trocando esta linha:

```
80  IF T = 45 THEN 200
```

CHECANDO SUAS RESPOSTAS

Como faria você para obrigar o computador a verificar suas respostas?

Será que isto funciona?

```
100 IF A$ = X + Y THEN 130
110 PRINT "ERRADO" X " + " Y "=" X + Y
120 GOTO 10
130 PRINT "CORRETO"
140 GOTO 10
```

Se executar este programa (e responder a tempo), você verá esta mensagem de erro:

Lembre-se do problema de misturar strings com números? Consulte o capítulo 2, ele refrescará sua memória.

```
?TM ERRO IN 100
```

Isto ocorre porque não é possível comparar A$, um string, com X + Y, um número. Você tem que converter A$ em um número.

Felizmente o computador sabe fazer isto. Mude a linha 100 para:

```
100 IF VAL (A$) = X + Y THEN 130
```

VAL (A$) transforma A$ em seu valor numérico. Se A$ é igual ao string "5"; VAL(A$) é igual ao número 5. Contudo, se VAL (A$) é igual ao string "C"; VAL (A$) é igual a 0 porque "C" não é um número.

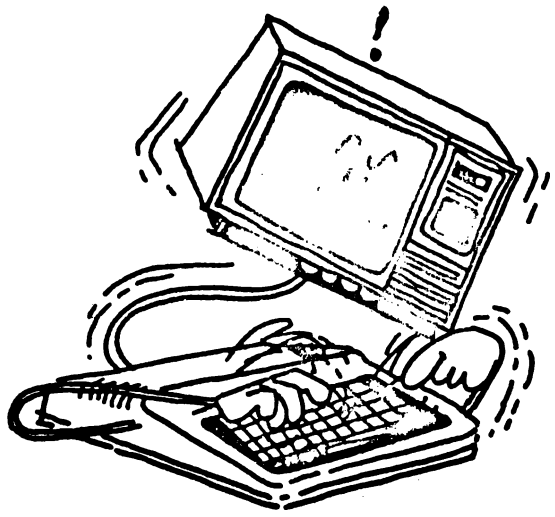
Para aqueles que querem ir um pouco mais longe, mudem estas linhas:

```
10  X = RND (49) + 4
20  Y = RND (49) + 4
90  B$ = B$ + A$
100 IF VAL(B$) = X + Y THEN 130
```

e acrescente estas linhas:

```
45  B$ = ""
95  IF LEN (B$) < > 2 THEN 50
```

UM TESTE DE DATILOGRAFIA



Eis um programa que fará o computador cronometrar sua velocidade de datilografia:

```
10  CLS
20  INPUT "PRESSIONE < ENTER > QUANDO ESTIVER PRONTO PARA DATILOGRAFAR ESTA
    SENTENCA"; E$
30  PRINT "AGORA E' TEMPO PARA TODOS OS HOMENS BONS"
40  T = 1
50  A$ = INKEY$
60  IF A$ = "" THEN 100
70  PRINT A$ ;
80  B$ = B$ + A$
90  IF LEN (B$) = 40 THEN 120
100 T = T + 1
110 GOTO 50
120 S = T/74
130 M = S/60
140 R = 8/M
```

```
150 PRINT
160 PRINT "VOCE DATILOGRAFOU - "R" - PALAVRAS/MIN"
```

Aqui está como funciona este programa:

Na linha 40 inicializamos o contador (T) com 1.

A linha 50 dá a primeira oportunidade para você pressionar uma tecla — A$ —. Se você não for rápido, a linha 60 manda o computador diretamente para a linha 100 e adiciona 1 ao contador.

A linha 70 imprime a tecla pressionada.

"Agora é tempo para todos os homens bons"

A linha 80 forma um string chamado B$. Cada vez que você pressionar uma tecla (A$), ela será somada a B$. Por exemplo: suponhamos que a primeira tecla seja "A", então:

```
A$ = "A"
```

e

```
B$ = B$ + A$
B$ = "" + "A"
B$ = "A"
```

Suponhamos que a próxima tecla seja "G", então:

A$ = "G"

e

B$ = B$ + A$
B$ = "A" + "G"
B$ = "AG"

Suponhamos que a terceira, quarta e quinta teclas sejam "O", "R" e "A" sucessivamente.

A$ = "O", "R" e "A" sucessivamente

e

B$ = "AGORA"

(formamos a palavra "AGORA")

Poderíamos ter feito este cálculo em apenas uma linha, usando parênteses:
120 R = 8/((T /74) /60)

Quando o comprimento de B$ for igual a 40 caracteres (a dimensão de "AGORA E' TEMPO PARA TODOS OS HOMENS BONS"), o computador sabe que você terminou a datilografia da sentença e vai para a linha 120 onde calcula o número de palavras datilografadas por minuto.

Que tal tentar uma variante deste programa? Um teste de velocidade de leitura, por exemplo.

Calculamos o número de palavras por minuto nas linhas 120, 130 e 140, dividindo T por 74 (para determinar os segundos), S por 60 (para calcular os minutos - M) e então dividimos 8 palavras por M para determinar o percentual de palavras por minuto.

Altere o programa para que o computador verifique se for cometido algum erro de datilografia.

FAÇA VOCÊ MESMO O SEU PROGRAMA

Nossa resposta está no apêndice I no fim deste manual.

ASSUNTOS APRENDIDOS NO CAPÍTULO 13

COMANDOS BASIC

INKEY$

VAL

CAPÍTULO 14

APERFEIÇOAMENTO

14

APERFEIÇOAMENTO

Apresentaremos agora alguns comandos BASIC que não são indispensáveis, mas o ajudarão a fazer programas mais facilmente.

O primeiro comando é STOP. Digite e execute este programa:


```
10  A = 1
20  A = A + 1
30  STOP
40  A = A*2
50  STOP
60  GOTO 20
```

O computador imprime:

```
BREAK IN 30
OK
```

O computador parou o programa quando atingiu a linha 30. Neste ponto podemos digitar uma linha de comando para ver o que o programa fez até aqui. Por exemplo, digite:

PRINT A (ENTER)

O computador imprime 2, o valor de A, quando ele parou o programa. Agora digite:

CONT (ENTER)

O computador continua a execução do programa a partir de onde parou. Em outras palavras, ele continua a execução do programa a partir da linha 40. A seguir ele imprime:

BREAK IN 50

Na linha 50 também existe um STOP. Digite novamente:

PRINT A (ENTER)

Ele imprime 4, o valor de A na linha 50. Digite CONT outra vez e o computador novamente para na linha 30. Se digitar PRINT A, ele imprime 5, que é o valor de A na segunda passagem pela linha 30.

STOP e CONT são comandos para serem usados quando seu programa não funcionar. Colocando STOP no programa, você pode analisar o que está errado. Uma vez localizado o problema, você pode tirar as linhas STOP.

PARA PROGRAMADORES AMBICIOSOS

Limpe a memória (NEW) e digite esta linha:

PRINT MEM (ENTER)

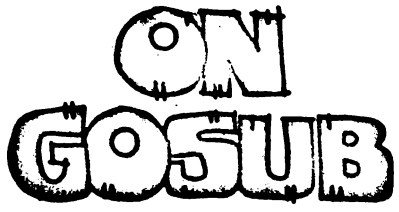
Ele imprime 31015 . Este número indica o tamanho da memória livre para armazenamento.

Para economizar memória, você pode omitir espaços no programa antes e depois de pontuações, operações e comandos BASIC.

Quando estiver digitando um programa longo, você poderá pedir que o computador imprima MEM de tempos em tempos para certificar-se que ainda existe memória disponível.

AJUDA NA DIGITAÇÃO

Digite este programa:




```
10   INPUT "DIGITE 1, 2, OU 3"; N
20   ON N GOSUB 100, 200, 300
30   GOTO 10

100  PRINT "VOCE DIGITOU 1"
110  RETURN

200  PRINT "VOCE DIGITOU 2"
210  RETURN
```

```
300  PRINT "VOCE DIGITOU 3"
310  RETURN
```

Execute.

A linha 20 poderia ser substituída por estas três:

```
18 IF N = 1 THEN GOSUB 100
20 IF N = 2 THEN GOSUB 200
22 IF N = 3 THEN GOSUB 300
```

Você reduz a quantidade de linhas usando ON ... GOSUB.

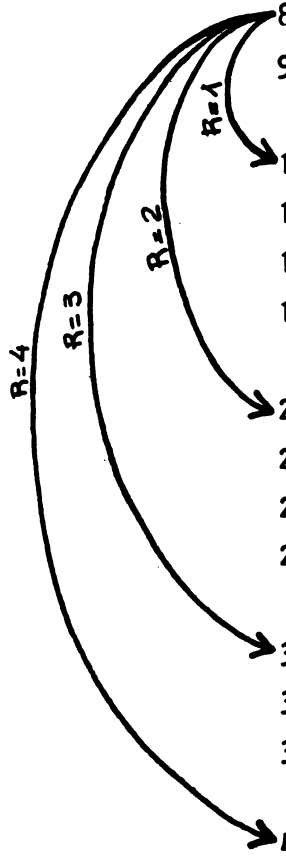
ON ... GOSUB diz ao computador para analisar o número que segue ON. Neste caso, o número é N. Se for 1, o computador vai para a subrotina que começa no primeiro número depois do GOSUB.

Se N é 2, o computador vai para a subrotina que começa no segundo número após o GOSUB . Se N é 3, o computador vai para a subrotina que começa no terceiro número após o GOSUB.

O que acontece se N for 4? Como não existe um quarto número após o GOSUB, o computador , simplesmente, vai para a linha imediatamente abaixo do ON ... GOSUB.

Aqui está um programa que usa ON ... GOSUB:

```
5   FOR P = 1 TO 600: NEXT P
```




```
10   CLS:X = RND(100):Y = RND(100)
20   PRINT "(1) ADICAO"
30   PRINT "(2) SUBTRACAO"
40   PRINT "(3) MULTIPLICACAO"
50   PRINT "(4) DIVISAO"
60   INPUT "QUAL A OPERACAO (1-4)"; R
70   CLS
80   ON R GOSUB 1000, 2000, 3000, 4000
90   GOTO 5

1000 PRINT "QUANTO E''' X "+" Y
1010 INPUT A
1020 IF A = X+Y THEN PRINT "CORRETO" ELSE PRINT "ERRADO"
1030 RETURN

2000 PRINT "QUANTO E''' X "-" Y
2010 INPUT A
2020 IF A = X-Y THEN PRINT "CORRETO" ELSE PRINT "ERRADO"
2030 RETURN

3000 PRINT "QUANTO E''' X "*" Y
3010 INPUT A
3020 IF A = X*Y THEN PRINT "CORRETO" ELSE PRINT "ERRADO"

4000 PRINT "QUANTO E''' X "/" Y
4010 INPUT A
```

```
4020 IF A = X / Y THEN PRINT "CORRETO" ELSE PRINT "ERRADO"
4030 RETURN
```

Quando A for diferente de X+Y, a condição da linha 1020 não é verdadeira.

Veja a palavra ELSE nas linhas 1020, 2020, 3020 e 4020. Você pode usar ELSE caso queira que o computador faça alguma coisa especial quando a condição não for verdadeira. Na linha 1020 se (IF) a resposta -A- for igual a X+Y, então (THEN), o computador imprime CORRETO senão (ELSE), imprime ERRADO.

Você pode usar ON ... GOTO do mesmo modo que ON ... GOSUB. A única diferença é que ON ... GOTO manda o computador para um outro número de linha, ao invés de uma subrotina.

Aqui está parte de um programa com ON ... GOTO:

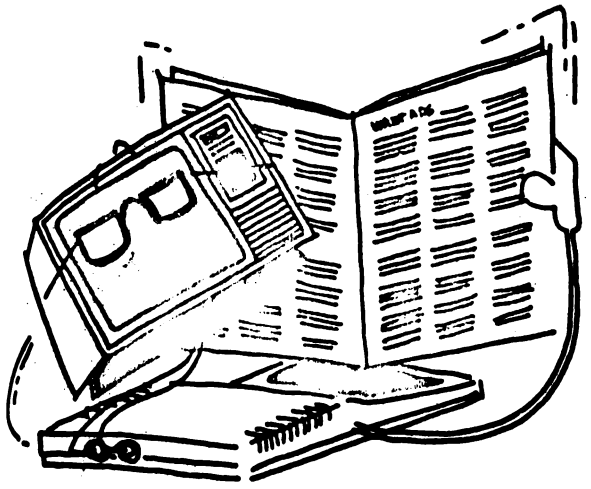

```
10   CLS
20   PRINT @ 134, "(1) ROLETA RUSSA"
30   PRINT @ 166, "(2) 21"
40   PRINT @ 198, "(3) CORACOES"
50   PRINT @ 354, "O QUE VOCE QUER JOGAR"
60   INPUT A
65   CLS
70   ON A GOTO 1000, 2000, 3000

1000 PRINT @ 230, "JOGO ROLETA RUSSA"
1010 END

2000 PRINT @ 236, "JOGO DE 21"
2010 END
```

A:1

A:2

```
3000 PRINT       235, "JOGO DE CORACOES"
3010 END
```

O TRABALHO DIZ "AND" OU "OR"?



Qualquer um conhece a diferença entre E e OU (AND e OR em inglês), até mesmo o computador. Por exemplo, digamos que uma empresa tenha uma série de vagas para programadores e que para conseguir o emprego você precisa ter:

Um diploma de programação

E

Experiência em programação

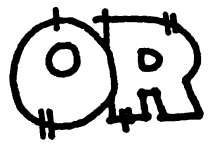
Com se diz isto em um programa? Apague a memória e digite:

```
10   PRINT "VOCE TEM--"
20   INPUT "UM DIPLOMA DE PROGRAMACAO"; D$
30   INPUT "EXPERIENCIA EM PROGRAMACAO"; E$
40   IF D$ = "SIM" AND E$ = "SIM" THEN PRINT "VOCE CONSEGUIU O EMPREGO"
     ELSE PRINT "DESCULPE, MAS NAO PODEMOS DAR-LHE O EMPREGO"
50   GOTO 10
```

Execute o programa. Com sua experiência em computador, você deve responder assim às perguntas:

```
VOCE TEM--
UM DIPLOMA DE PROGRAMACAO? NAO
EXPERIENCIA EM PROGRAMACAO? SIM
DESCULPE, MAS NAO PODEMOS DAR-LHE O EMPREGO
```

Suponhamos que esta empresa decidiu ser mais camarada. Aqui está a nova qualificação profissional necessária:



Um diploma de programação

OU

Experiência em programação

Tudo que foi feito foi uma mudança de palavra. Foi trocado o E por OU. Para fazer isto, mude seu programa e digite:

```
40 IF D$ = "SIM" OR E$ = "SIM" THEN PRINT "VOCE CONSEGUIU O EMPREGO"  ELSE
   PRINT "DESCULPE, MAS NAO PODEMOS DAR-LHE O EMPREGO"
```

Para ver a diferença que esta palavra faz, execute o programa:

```
VOCE TEM--
UM DIPLOMA DE PROGRAMACAO? NAO
EXPERIENCIA EM PROGRAMACAO? SIM
VOCE CONSEGUIU O EMPREGO
```

Agora sabendo que o computador compreende o sentido de AND e OR, você pode usá-los nos programas. Usaremos estas palavras em outras oportunidades.

EXISTE MAIS AJUDA PARA A MATEMÁTICA

Existem outras palavras que podem ser usadas para ajudá-lo com programas matemáticos:

SGN

SGN diz se o número é positivo, negativo ou zero.

Digite:

```
10   INPUT "DIGITE UM NUMERO"; X
20   IF SGN(X) = 1 THEN PRINT "POSITIVO"
30   IF SGN(X) = 0 THEN PRINT "ZERO"
40   IF SGN(X) =-1 THEN PRINT "NEGATIVO"
```

Execute o programa. Tente introduzir estes números:

```
15    -30    -.012    0    .22
```

ABS

ABS fornece o valor absoluto de um número. Digite:

```
10   INPUT "DIGITE UM NUMERO"; N
20   PRINT "O VALOR ABSOLUTO E'" ABS(N)
30   GOTO 10
```

Execute o programa, introduzindo os números anteriores.

STR$

STR$ converte um número para um string. Exemplo: digite e execute:

```
10   INPUT "DIGITE UM NUMERO"; N
20   A$ = STR$(N)
30   PRINT A$ + "AGORA E' UM STRING"
```

OVERFLOW

Digite e execute este programa:

Observe o OV ERROR (overflow) no final. O computador não pode manipular números maiores que 1E + 38 ou menores que -1E + 38.

```
10   X = 1
20   PRINT X;
30   X = X * 10
40   GOTO 20
```

Algumas vezes um número ficará tão grande, ou então tão pequeno que o computador o imprimirá em "notação científica". O número um bilhão (1.000.000.000), por exemplo, pode ser escrito como "1E + 09". Isto significa "o número 1 é seguido por nove zeros".

*Ou tecnicamente, $1 * 10^9$, que é 10 elevado a 9. 1*10*10*10*10*10*10*10*10*10*

Se a resposta for "5E - 06", isto significa que devemos deslocar o ponto decimal que segue o número 5, seis casas para a esquerda, inserindo tantos zeros quantos forem necessários. Tecnicamente, isto significa $5 * 10^{-6}$, ou (0.000005). Isto é realmente muito simples depois que você entender. Explicamos este assunto rapidamente, mas achamos que você apreciará estas informações quando estiver praticando seus próprios programas.

No BASIC é: 5/10/10/10/10/10/10

A S S U N T O S   A P R E N D I D O S   N O   C A P Í T U L O   1 4

COMANDOS BASIC		SÍMBOLOS BASIC	CONCEITOS BASIC
STOP	SGN	AND	Notação científica
CONT	ABS	OR	
MEM	STR$		

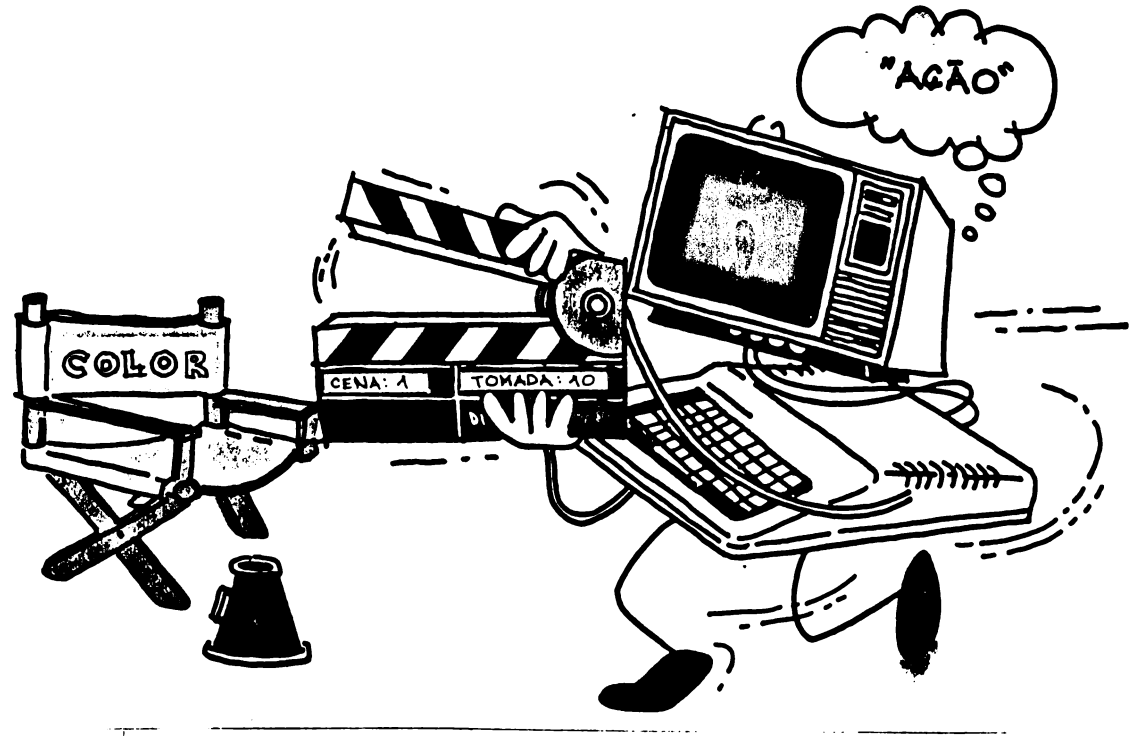
SEÇÃO II

GRÁFICOS ANIMADOS

Aqueles que desejarem escrever programas cheios de cor e emoção, gostarão certamente desta seção. Nela faremos desenhos que se movem, dançam e até mesmo falam.

Para tornar as coisas simples, nossos programas serão bem curtos. Uma vez que entenda o que está fazendo, você poderá criar facilmente os seus próprios programas com muito mais atrativos.

CAPÍTULO 15



DESENHOS ANIMADOS

15

DESENHOS ANIMADOS

Que tal dar um pouco de vida aos seus programas? A animação é o recurso que torna os gráficos mais divertidos.

Existem duas maneiras de programar gráficos. A primeira é usando os comandos SET e RESET para posicionar um ponto na tela. É este método que adotaremos primeiro. No capítulo 18 mostraremos o outro método.

Digite:

```
PRINT ASC("A") (ENTER)
```

e o computador imprime:

```
65
```

65 é o código ASCII do caráter A. Digite:

```
PRINT CHR$(65) (ENTER)
```

e o computador imprime A, mostrando que o caráter A tem o código ASCII número 65.

"ASCII" é a sigla de American Standard Code for Information Interchange. Com o uso deste código, o computador é capaz de se comunicar com outros computadores pelo telefone.

Veja a lista de "Códigos de Caracteres ASCII" no apêndice E. Cada caráter do teclado possui um código. Experimente outros caracteres...

Como isto pode ajudar na elaboração de gráficos? Para a maioria dos caracteres do teclado você pode programar o que o computador deve fazer quando eles forem pressionados. Por exemplo, você pode digitar estas linhas no seu programa:

```
10 INPUT A$
20 IF A$ ="W" THEN H = H-1
```

para dizer ao computador o que fazer quando for digitado o caráter W.

Falaremos sobre alguns outros usos de CHR$ mais adiante nesta seção.

Entretanto, se tentar substituir a tecla W pela tecla (←), na linha 20, o computador não permitirá que você o faça. Isto ocorre porque o computador já decidiu o que fazer quando for digitada a tecla (←).

Para ilustrar melhor o fato, podemos usar CHR$(8) para representar o caráter (←). Limpe (NEW) a memória do computador e digite:

```
10 CLS(0)
20 H = 63
25 SET (H,14,3)
```

```
30 A$ = INKEY$
40 IF A$ = CHR$(8) THEN 60
50 GOTO 30
60 H = H-1
65 IF H < 0 THEN END
70 SET (H,14,3)
75 RESET (H+1,14)
80 GOTO 30
```

Para rever o comando INKEY$, consulte o capítulo 13.

Execute o programa. Pressione o caráter (←). Toda vez que ele é pressionado, o ponto azul anda uma posição para trás.

A linha 30 diz ao computador para chamar de A$ a tecla pressionada. Se A$ for igual ao caráter representado pelo código 8 - o caráter (←) - então o computador desviará para a linha 60.

Nas linhas 60 e 70, a coordenada horizontal H é retrocedida uma posição e um ponto azul se acende (SET). Em seguida, na linha 75, o ponto azul anterior é apagado.

Escreva algumas linhas a mais de forma que ao pressionarmos o caráter (→), o computador avance o ponto numa posição.

FAÇA VOCÊ MESMO O SEU PROGRAMA

Nossa resposta encontra-se no apêndice I no fim deste manual.

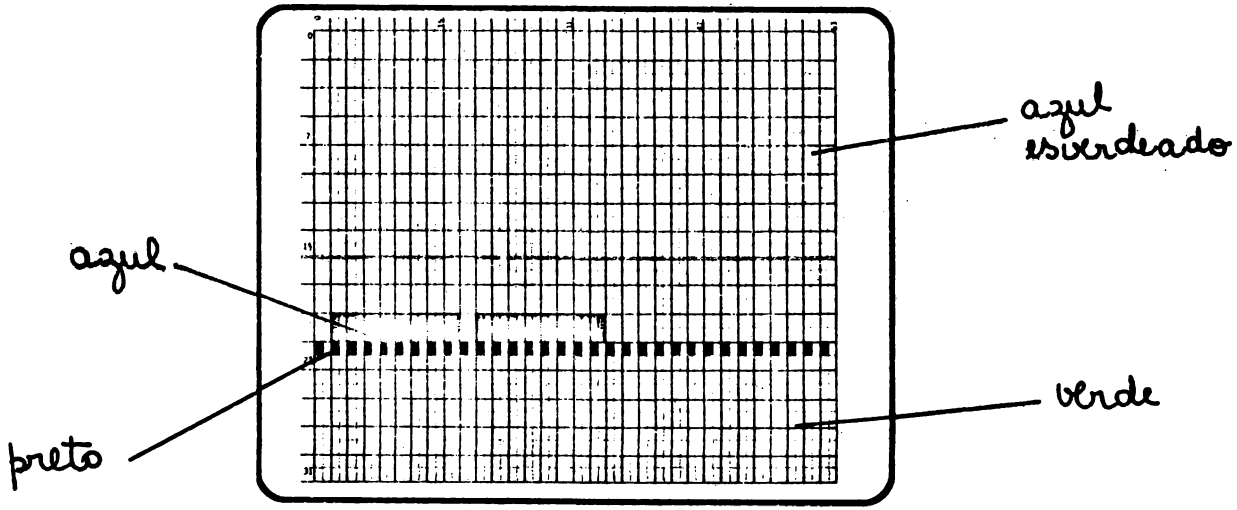
UM TREM QUE SE MOVE

Agora que você entendeu a função de CHR$, você pode usá-lo em desenhos animados.

Antes de escrever o programa, vamos fazer um esboço da aparência do desenho, usando o gabarito do apêndice D, "Posicionamento de Gráficos".

Nossa cena vai ter a seguinte aparência:

É muito mais fácil programar usando o gabarito para planejar os gráficos com antecedência. Use um lápis ou tire algumas cópias do gabarito, para não inutilizá-lo



Para rever este assunto, consulte o capítulo 9.

Repare como o gabarito está dividido em blocos - os quadros delineados por linhas espessas. Cada bloco contém 4 pontos que devem estar:

- todos com a mesma cor
- uma cor e preto

Como o desenho dos trilhos é preto, podemos deixar que compartilhe o mesmo bloco da grama verde.

Primeiro vamos criar o nosso cenário. Depois de digitar as linhas para criar cada parte do cenário - o céu, a grama, os trilhos e o trem - você pode executar o programa para ver como ficou.

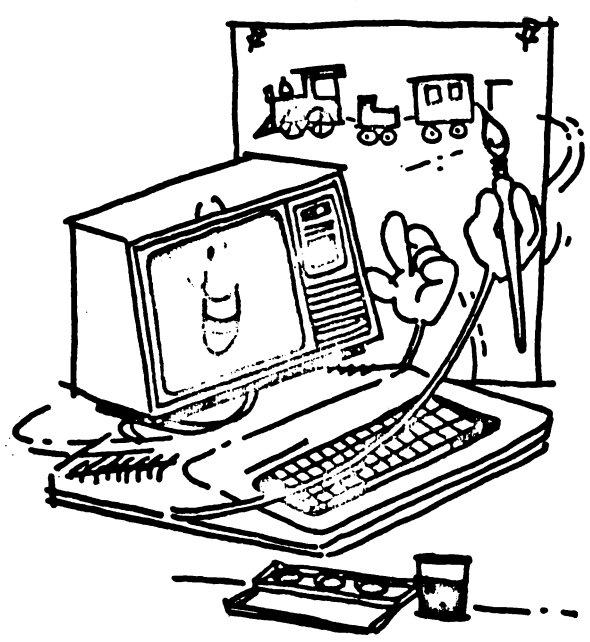
Pressione (BREAK) para parar o programa.

Para tornar o céu azul esverdeado, limpe a memória e digite:

```
10 CLS(6)
```

Para tornar a grama verde, digite:

```
20 FOR H =  0 TO 63
30 FOR V = 22 TO 31
40 SET (H,V,1)
50 NEXT V,H
```



Isto faz com que todos os pontos fiquem verdes (cor nº 1) desde a posição 0 até a 63 na horizontal, e desde a posição 22 até a 31 na vertical. Repare que a linha 50 contém, na verdade, duas instruções:

```
NEXT V
NEXT H
```

Para desenhar os trilhos, digite:

```
60 FOR H =  0 TO 63 STEP 2
70 RESET (H,22)
80 NEXT H
```

Isto faz com que os pontos parés da linha 22 se tornem pretos.

Como não colocamos nenhum comando GOTO para deixar o programa em loop, a tela apresentará uma faixa verde no topo com a mensagem OK.

Para desenhar o trem, digite:

```
90  FOR V = 20 TO 21
100 FOR H =  0 TO 15
110 SET (2+H,V,3): SET(20+H,V,3)
120 NEXT H,V
```

Isto desenhará uma composição de dois vagões, cada um com um comprimento de 16 pontos (0 a 15).O primeiro começa na posição horizontal 2 e o segundo na posição 20.

Execute o programa para ter certeza que o seu cenário se parece com o que esboçamos acima.

Está tudo de acordo? Agora vamos fazer o trem andar para frente. Digite:

```
200 A$ = INKEY$
210 IF A$ = CHR$(9) THEN GOSUB 1000
220 GOTO 200
```

Isto simplesmente diz ao computador que caso você pressione a tecla (→) - o caráter representado pelo código 9 - então o computador irá para uma subrotina começando na linha 1000. Aqui está a subrotina. Digite:

```
1000 REM PARA FRENTE
1010 IF F > 26 THEN RETURN
1020 FOR V = 20 TO 21
1030 FOR H =  0 TO  1
```

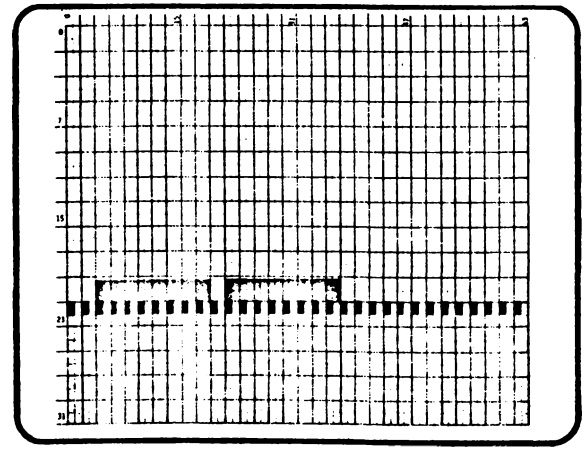
```
1040 SET(2+F+H,V,6):SET(20+F+H,V,6)
1050 SET(18+F+H,V,3):SET(36+F+H,V,3)
1060 F = F+2
1070 GOTO 1000
```

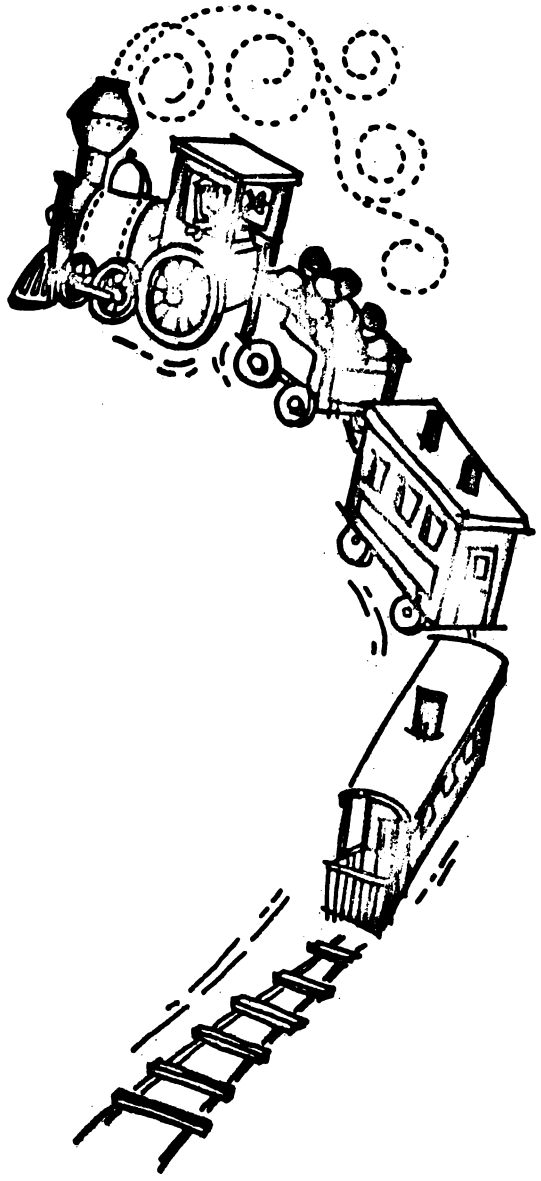
Neste ponto F=0.

Execute o programa. Pressione a tecla (➡) e o trem se moverá para a frente.

A linha 1040 coloca a cor azul no primeiro bloco de cada vagão. (os blocos começando nas posições 2 e 20.)

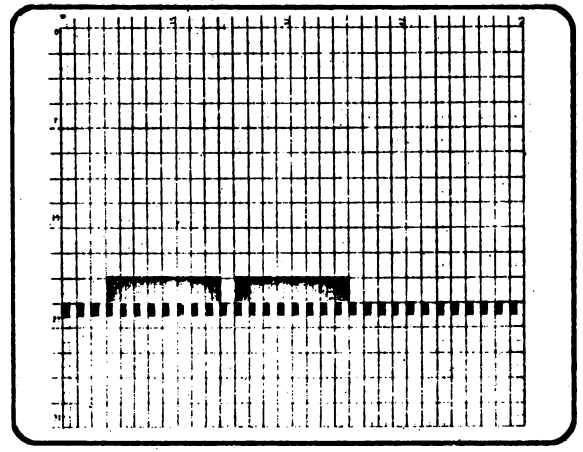
A linha 1050 coloca a cor do trem no primeiro bloco à frente de cada vagão (os blocos que começam nas posições 18(2+16) e 36(20+16). Depois que o computador executar a linha 1050, a tela ficará da seguinte forma:





Veja como cada vagão se moveu um bloco para frente. É claro que esta imagem só durará um instante, pois a linha 1060 faz F=2 e a linha 1070 desvia o computador para o início da rotina novamente.

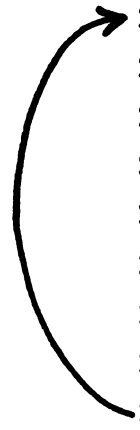
Na segunda passagem pela rotina, os blocos das posições 4(2+F) e 22(20+F) são acesos na cor do céu e os blocos das posições 20(18+F) e 38(36+F) são acesos na cor do trem. Isto faz o trem se mover mais um bloco à frente.



O trem se movimenta, bloco a bloco, até atingir o fim da tela. Isto ocorre quando F = 26 . Depois disto, a linha 1010 faz o computador retornar para a linha 220.

Você quer fazer o trem andar para trás? Acrescente estas linhas:

```
215  IF A$ = CHR$(8) THEN GOSUB 2000
```



```
2000 REM PARA TRAS
2010 IF F < 0 THEN RETURN
2020 FOR V = 20 TO 21
2030 FOR H =  0 TO  1
2040 SET(0+F+H,V,3):SET(18+F+H,V,3)
2050 SET(16+F+H,V,6):SET(34+F+H,V,6)
2060 NEXT H,V
2070 F = F-2
2080 GOTO 2000
```

Execute o programa. Pressione a tecla (←) e o trem andará para trás. Pressione a tecla (→) e ele andará para frente.

O método que acabamos de apresentar funciona muito bem com imagens pequenas e simples. Entretanto, se você quiser movimentar uma imagem mais complexa, é melhor usar o método que mostraremos no capítulo 18.

ASSUNTOS APRENDIDOS NO CAPÍTULO 15

PALAVRAS BASIC

ASC

CHR$

CAPÍTULO 16

O PROFESSOR QUE FALA

16

O PROFESSOR QUE FALA

Quem disse que um computador não pode falar? Ele fala, mas, sua voz, entretanto soa estranha, parecida com a sua própria...

Podemos fazer o computador falar, usando para tal a sua própria voz gravada numa fita cassete. Com isto, você pode aumentar sensivelmente o interesse e a animação dos seus programas (principalmente em jogos e programas educativos). Mesmo que não tenha um gravador, você poderá aproveitar algumas idéias apresentadas neste capítulo.

Desconecte o cabo que liga o computador ao gravador cassete. Instale uma fita, rebobine-a até o início, pressione as teclas PLAY e RECORD e fale no microfone (instale um microfone caso seu gravador não possua um).

Mesmo que não tenha um microfone, você pode fazer este programa usando uma fita pré-gravada com músicas ou programas.

Agora digite este programa:

```
5  CLS
10 INPUT "PRESSIONE <ENTER> PARA OUVIR A GRAVACAO"; A$
20 MOTOR ON
30 AUDIO ON
```

Pronto? Antes de executar o programa, você tem que preparar a fita para reprodução:

- rebobine a fita até o início da gravação
- conecte o gravador cassete ao computador
- pressione a tecla PLAY do gravador
- aumente o volume da televisão

O capítulo 8 ensina como conectar o gravador cassete.

Execute o programa. Você ouvirá a sua voz na televisão.

MOTOR ON aciona o gravador. AUDIO ON conecta o som do gravador ao auto-falante da televisão.

Existe uma maneira de programar que faz parar o seu gravador, mas, por enquanto, pressione simplesmente o botão RESET que se encontra na parte de trás do computador à direita de quem olha para ele. Veja (LIST) o seu programa, ele ainda está intacto.

Acrescente as linhas abaixo:

```
35 CLS
40 A$ = INKEY$
50 PRINT @ 224,"PRESSIONE <X> PARA DESLIGAR O GRAVADOR"
60 IF A$<>"X" THEN 40
70 AUDIO OFF
80 MOTOT OFF
```

Prepare o seu gravador para reprodução e execute (RUN) o programa.

A linha 40 diz ao computador para chamar de A$ qualquer tecla pressionada ou não. Se você não estiver pressionando um "X", a linha 60 desvia o controle do programa para a linha 40. Se você pressionar um "X", as conexões de AUDIO e MOTOR são desligadas.

Agora que você compreende como isto funciona, você está preparado para gravar o "PROFES - SOR QUE FALA". Espere um momento, aqui está o roteiro:

R O T E I R O

"Olá, eu sou o professor que fala. A primeira lição que faremos será de Matemática. Eu vou lhe dar uma série de operações de adição. Pressione a tecla 'E'..."

(pausa de alguns segundos)

"você vai ouvir isto todas as vezes que fornecer uma resposta Errada. Pressione a tecla 'C'..."

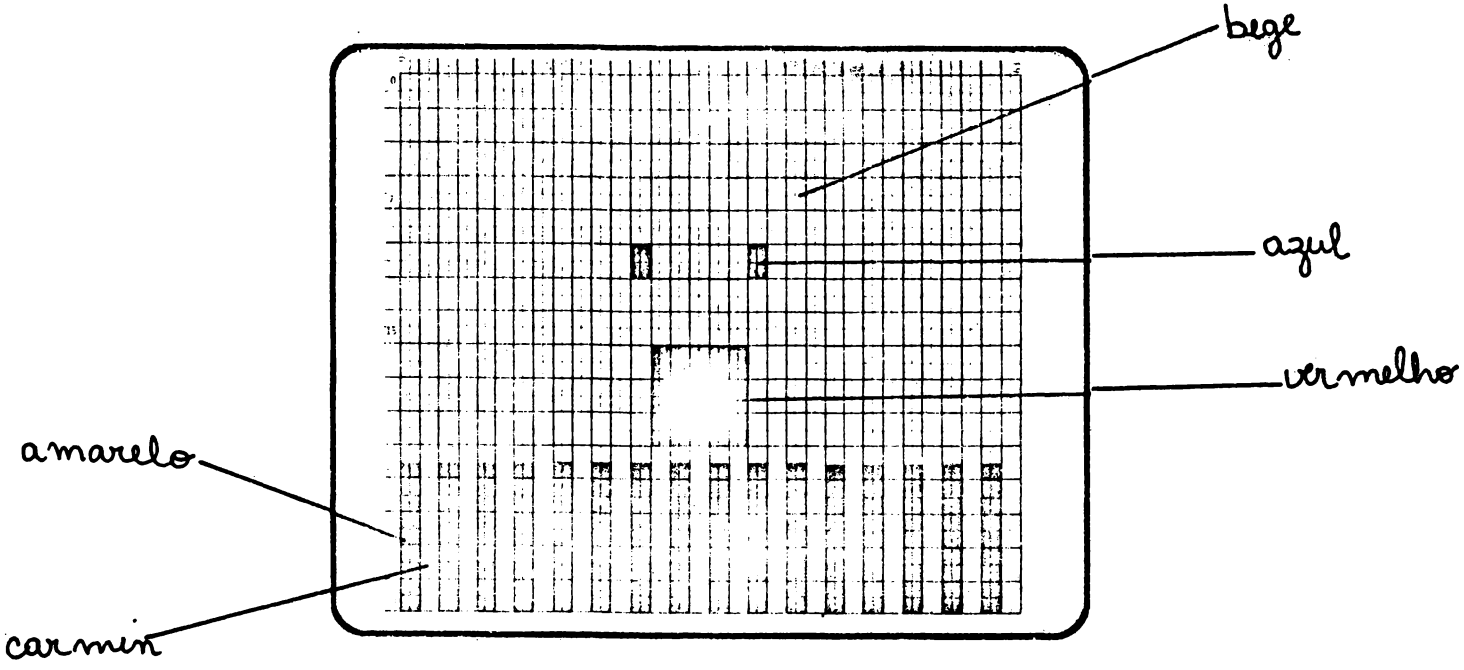
(pausa de alguns segundos)

"esta será a sua recompensa sempre que responder Corretamente. Eu não vou falar com você de novo, até que forneça três respostas corretas. Pressione a tecla 'I' para Iniciar."

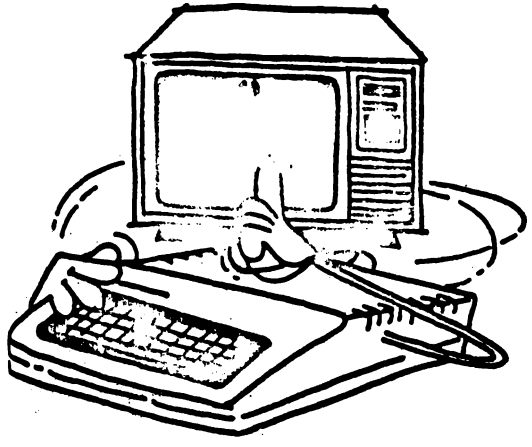
(pausa de alguns segundos)

"Olá novamente! Está na hora de começarmos a próxima lição, mas, como ainda não preparei esta lição, vou terminar o programa. Pressione a tecla 'D' para Desligar o gravador."

Terminou? A próxima coisa a fazer é desenhar o professor que fala. Aqui está um esboço de como ele parece:



Este programa é um pouco longo, mas acreditamos que você vai gostar dele. Se você quiser, pode ir para o próximo capítulo e voltar a este mais tarde.



Desenhe a boca primeiro. Limpe a memória e digite:

```
5   CLS(0)
200 FOR H = 26 TO 35
210 FOR V = 16 TO 21
220 SET (H,V,4)
230 NEXT V,H
```

Isto é uma boca fechada. Para fazê-la falar, digite:

```
500 RESET (30,18):RESET(30,19)
510 GOTO 200
```

e execute o programa. Não é uma boca muito bonita, mas, resolve o problema. Agora desenhe o rosto. Digite:

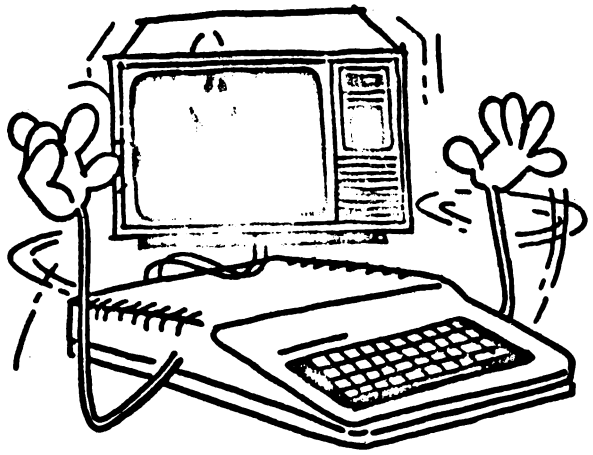
> *Lembre-se: você sempre pode pressionar o botão RESET para parar o gravador quando ele estiver conectado ao computador.*

```
100 FOR H = 16 TO 47
110 FOR V =  4 TO 23
120 SET (H,V,5)
130 NEXT V,H
```

e desenhe o corpo. Digite:

```
140 FOR H =  0 TO 63 STEP 4
150 FOR V = 24 TO 31
160 SET (H,V,2): SET (H+1,V,2)
170 SET (H+2,V,7): SET (H+3,V,7)
180 NEXT V,H
```

e os olhos. Digite:



```
300 FOR V = 10 TO 11
310 SET (24,V,3): SET (25,V,3)
320 SET (36,V,3): SET (37,V,3)
330 NEXT V
340 PRINT @ 6,"O PROFESSOR QUE FALA"
```

execute-o agora. Quer fazer os olhos piscarem? Digite:

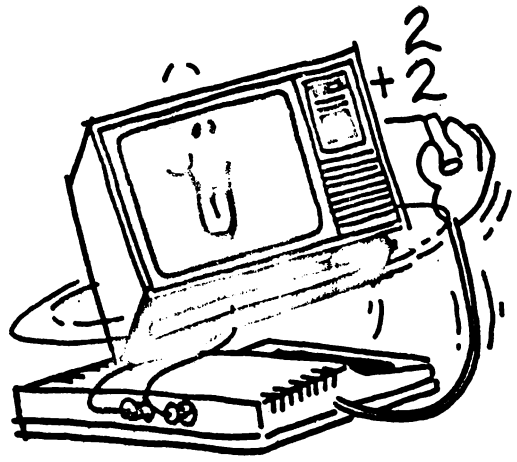
```
505 IF RND(4) = 4 THEN SET(24,10,5):SET(37,10,5)
```

e execute o programa. Este é o professor que fala.

Agora, faça-o falar. Digite:

```
400 MOTOR ON
410 AUDIO ON
420 A$ = INKEY$
430 IF A$ = "I" THEN MOTOR OFF : END
440 IF A$ = "E" THEN MOTOR OFF : GOSUB 2000
450 IF A$ = "C" THEN MOTOR OFF : GOSUB 3000

2000 FOR T = 176 TO 89 STEP -10
2010 SOUND T,1
2020 NEXT T
2030 RETURN

3000 FOR T = 89 TO 176 STEP 10
3010 SOUND T,1
3020 NEXT T
3030 RETURN
```

Antes de executar este programa, prepare a sua fita para reprodução (rebobine-a, conecte



o gravador ao computador e pressione a tecla PLAY). Agora execute o programa... Faça o que a voz lhe disser.

Ainda está trabalhando? Ao pressionar 'E' você deve ouvir tons descendentes; ao pressionar 'C' você deve ouvir tons ascendentes e 'I' simplesmente termina o programa. Isto ocorre porque você ainda não digitou a rotina aritmética.

Altere a linha 430 e acrescente a linha 460:

```
430 IF A$ = "I" THEN MOTOR OFF : GOSUB 1000
460 IF A$ = "D" THEN MOTOR OFF : END
```

e acrescente a rotina aritmética:

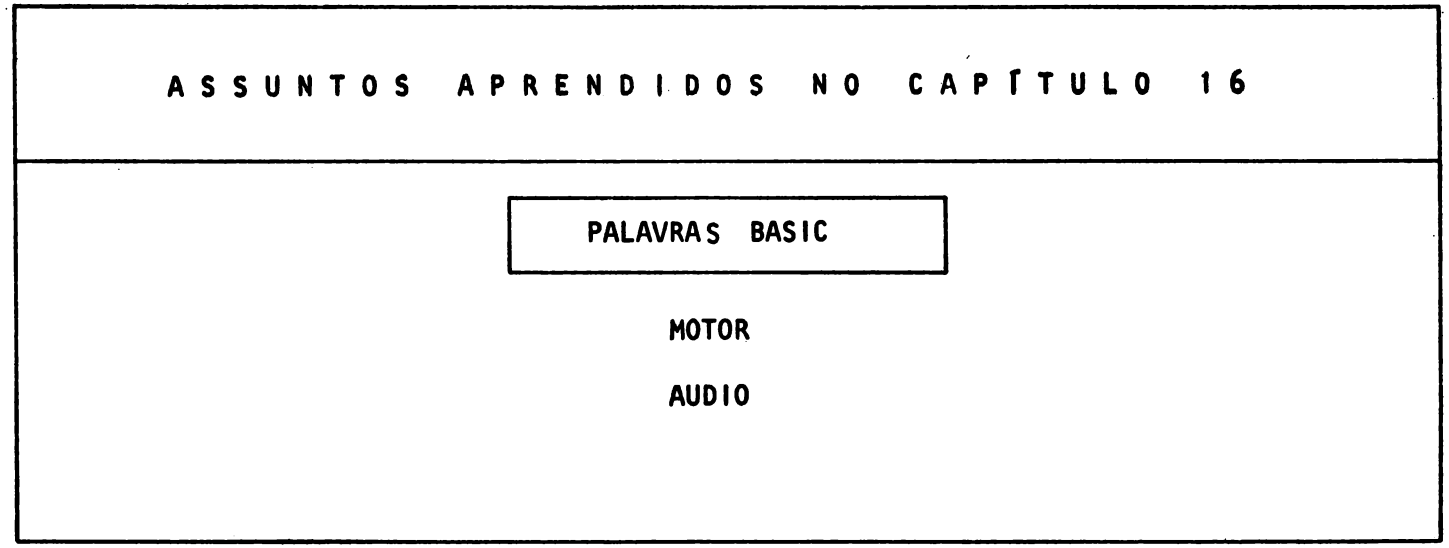
```
1000 X = RND(100)  : Y = RND(100)
1010 PRINT@ 0, "QUANTO E''' X "+" Y
1015 PRINT@ 20,
1020 INPUT A
1030 IF A = X+Y THEN GOSUB 3000 : C = C+1
1040 IF A< >X+Y THEN GOSUB 2000 : PRINT @ 0,"ERRADO. A RESPOSTA E''' X+Y
1050 IF C = 3 THEN RETURN
1060 FOR P = 1 TO 500 : NEXT P
1070 GOTO 1000
```

Note a linha 1015, ela ajusta a posição de impressão para os dados que você digitou na linha 1020.

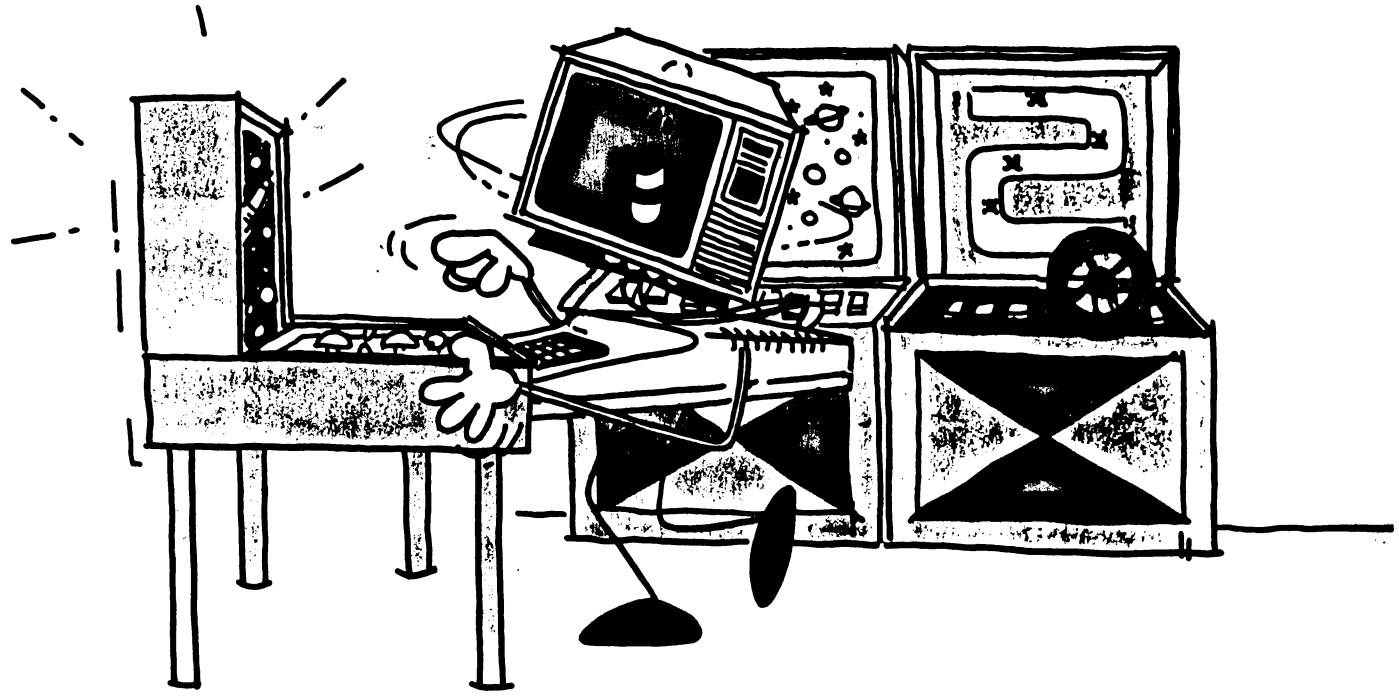
Rebobine a fita e pressione PLAY. Execute o programa...

Aí está; o professor que fala. Está perfeito para tornar o aprendizado da aritmética mais

divertido.


ASSUNTOS APRENDIDOS NO CAPÍTULO 16
PALAVRAS BASIC
MOTOR
AUDIO

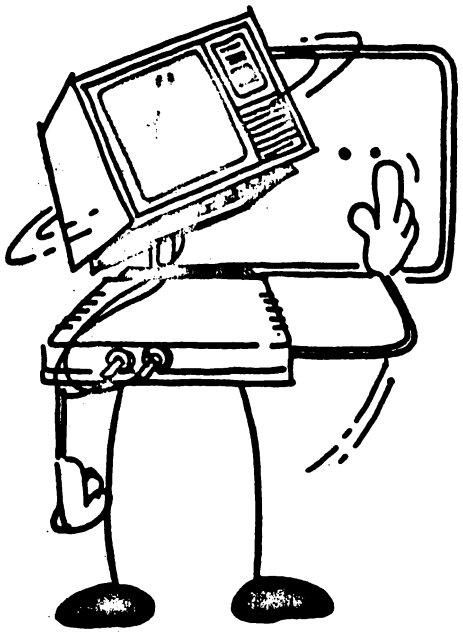
CAPÍTULO 17



JOGOS DE MOVIMENTO

17

JOGOS DE MOVIMENTO



Você está pronto para jogar um pouco de vídeo-tênis? Que tal brincar de tiro ao alvo ou de jogos espaciais? Você pode ensinar ao computador a jogar qualquer um destes jogos, assim que aprender mais uma palavra BASIC. Esta palavra é POINT e este capítulo é inteiramente dedicado a ela.

Limpe a memória e digite este programa:

```
 5  CLS(0)
10  FOR X = 1 TO 5
20  SET(RND(64)-1,RND(30)+1,8)
30  NEXT X
40  FOR V = 2 TO 31
50  FOR H = 0 TO 63
60  IF POINT(H,V)<>0 THEN GOSUB 100
70  NEXT H,V
80  END
```

```
100 PRINT @ 0,"A POSICAO " H "," V "ESTA' ACESA"
110 RETURN
```

Na linha 60, o computador varre cada ponto (POINT) na vertical, da posição 2 até a posição 31 e na horizontal da posição 0 até a 63, para verificar se ele está aceso. Se o ponto estiver aceso (ou seja, o ponto não é igual a zero) a linha 100 imprime as posições horizontal e vertical.

Elimine as linhas 40, 50 e 70 e altere a linha 10. Digite:

```
40
50
70
10 FOR X = 1 TO 300
```

Agora modifique as linhas 60 e 100, de forma que se o ponto (POINT) na posição 63,31 estiver aceso (SET), o computador imprima uma mensagem.

EXERCÍCIO DE PROGRAMAÇÃO

Aqui está como fizemos:

```
60  IF POINT(63,31)< >0 THEN GOSUB 100
100 PRINT@0,"A POSICAO 63,31 ESTA' ACESA"
```

Limpe a memória e digite este programa:

```
5  CLS(0)
10 C = RND(9)-1
20 SET(31,15,C)
30 IF POINT(31,15)=2 THEN PRINT@0,"A POSICAO 31,15 ESTA' AMARELA";
40 IF POINT(31,15)=3 THEN PRINT@480,"A POSICAO 31,15 ESTA' AZUL";
50 FOR T = 1 TO 1000 : NEXT T
60 GOTO 5
```

Execute-o e observe a tela por alguns instantes. POINT não testa apenas se um determinado ponto da tela está aceso, ele também verifica qual a sua cor. POINT será igual a zero se o ponto estiver apagado, ou igual ao código de uma das cores (consulte o apêndice B) se ele estiver aceso.

Acrescente duas linhas ao programa, para que o computador verifique se o ponto na posição 31,15 é verde ou vermelho.

EXERCÍCIO DE PROGRAMAÇÃO

Aqui estão estas duas linhas:

```
43 IF POINT(31,15)=1 THEN PRINT@160,"A POSICAO 31,15 ESTA' VERDE";
45 IF POINT(31,15)=4 THEN PRINT@320,"A POSICAO 31,15 ESTA' VERMELHA";
```

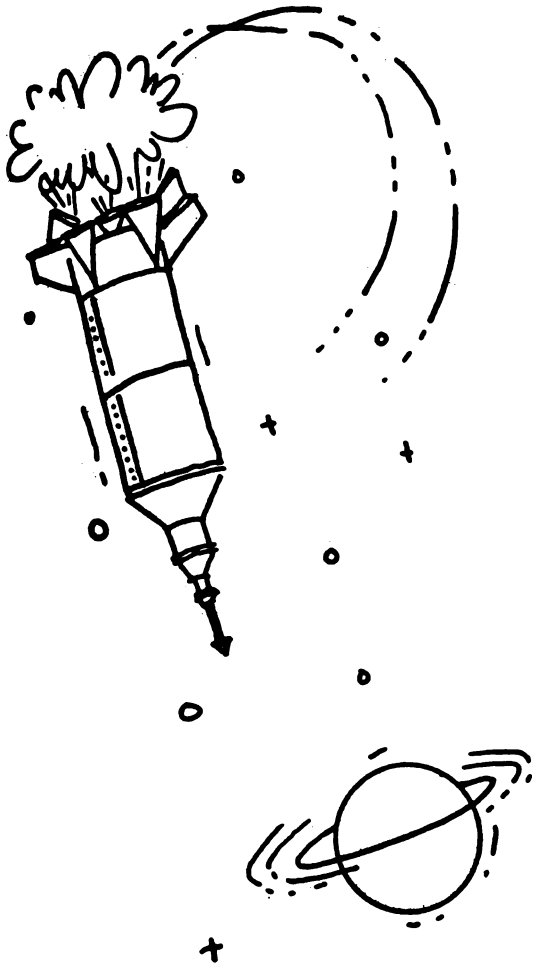
VIAJANDO EM MEIO A ASTERÓIDES

Neste jogo usaremos o joystick direito e por isso, verifique se ele está conectado.

Podemos criar asteróides da mesma forma que acendemos os pontos aleatórios acima. Limpe a memória e digite:

```
5 CLS(0)
```

```
10 FOR X = 1 TO 200
20 SET(RND(64)-1, RND(30)+1,8)
30 NEXT X
```



Para criar o planeta que a sua nave tem que alcançar, digite:

```
40 FOR H = 54 TO 63
50 FOR V = 28 TO 31
60 SET(H,V,3)
70 NEXT V,H
```

Para ler a posição do joystick direito, digite:

```
100 A = JOYSTK(0)
110 B = JOYSTK(1)
120 B = B/2
130 B = INT(B)
```

A lê as coordenadas horizontais (0-63) e B lê as verticais (0-63). Como a posição vertical mais alta na tela é 31, temos que acrescentar as linhas 120 e 130.

Para acender todo o bloco em torno da posição do joystick, acrescente estas linhas:

```
200 IF INT(A/2)< >A/2 THEN A=A-1
210 IF INT(B/2)< >B/2 THEN B=B-1
220 FOR H = A TO A+1
```

```
230 FOR V = B TO B+1
240 SET(H,V,6)
250 NEXT H,V
999 GOTO 100
```

As linhas 200 e 210 garantem que o primeiro ponto aceso na horizontal e na vertical tem coordenadas pares; as linhas 220 até 250 acendem todo o bloco.

Execute o programa e movimente o seu joystick. A linha de cor azul esverdeado acompanhará todos os movimentos do joystick.

Agora faça disto um jogo. Digite:

```
212 FOR H = A TO A+1
214 FOR V = B TO B+1
216 IF POINT(H,V)=8 THEN SOUND 128,1: T=T+1
218 NEXT V,H
```

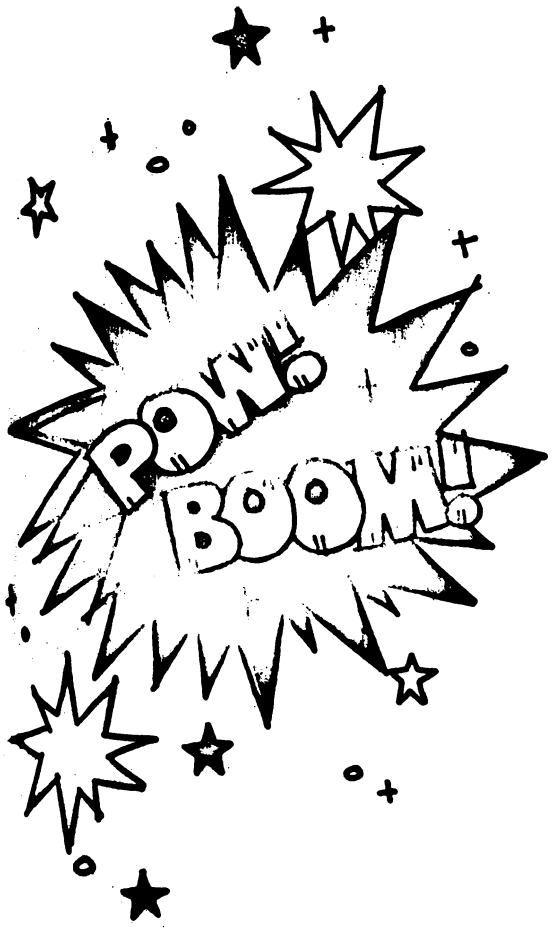
Execute novamente. Cada vez que você atingir um ponto laranja, o computador soará um tom.

Repare que a linha 216 faz duas coisas se o ponto for laranja:

- soa um tom
- acrescenta 1 ao contador (T)

Acrescente estas linhas ao seu programa:

```
235 IF POINT(H,V)=3 THEN PRINT@0,"PARABENS, VOCE CONSEGUIU":END
300 PRINT@ 28,T
310 IF T > 10 THEN 1000
1000 FOR X = 1 TO 40
1010 CLS(RND(8))
1020 SOUND RND(255),1
1030 NEXT X
1040 PRINT@228,"A SUA NAVE EXPLODIU"
```



e execute-o... você gostaria de ter instruções impressas na tela? Acrescente estas linhas:

```
80 FOR X = 1 TO 8
82 READ A$
84 PRINT@0,A$
86 FOR Y = 1 TO 1500 : NEXT Y
88 NEXT X
90 R$=INKEY$ : IF R$= "" THEN 90
92 FOR H = 4 TO 63
94 SET(H,0,8) : SET(H,1,8)
96 NEXT H
```

```
2000 DATA O SEU OBJETIVO E' TRACAR UM RUMO
2010 DATA PARA DIRIGIR A SUA NAVE
2020 DATA EM MEIO AOS ASTEROIDES
```

```
2030 DATA ATE' O PLANETA AZUL
2040 DATA SE VOCE BATER EM MAIS DE 10 ASTEROIDES,
2050 DATA A SUA NAVE ESPACIAL EXPLODIRA'.
2060 DATA PRESSIONE QUALQUER TECLA QUANDO A SUA
2070 DATA NAVE ESTIVER NO CANTO SUPERIOR ESQUERDO
```

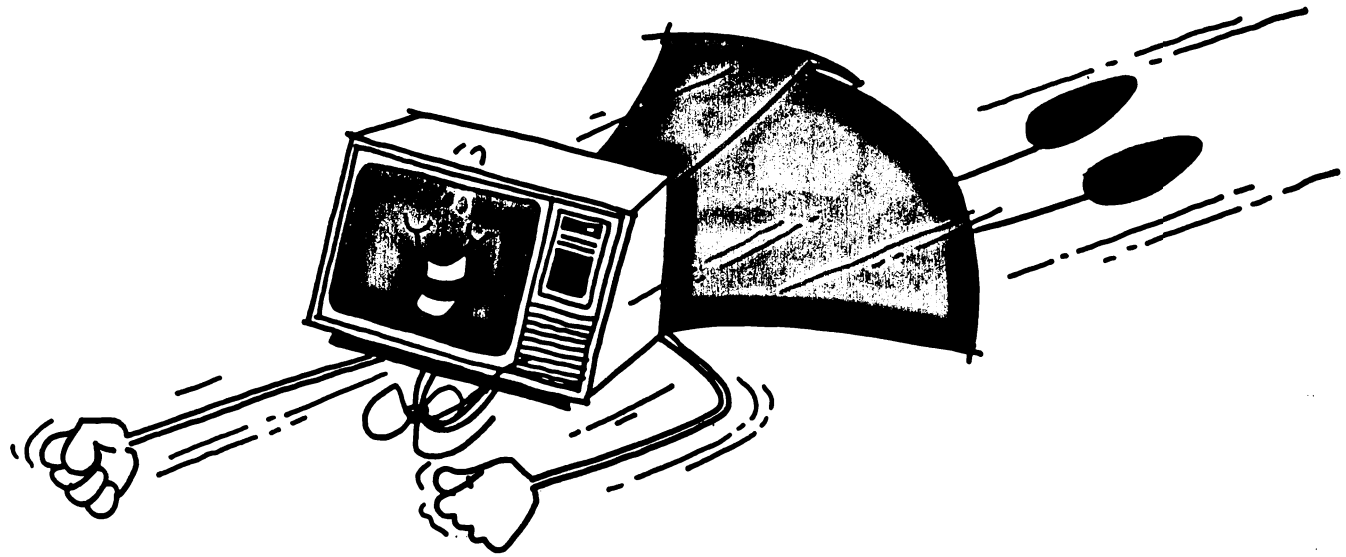
ASSUNTOS APRENDIDOS NO CAPÍTULO 17

PALAVRAS BASIC

POINT

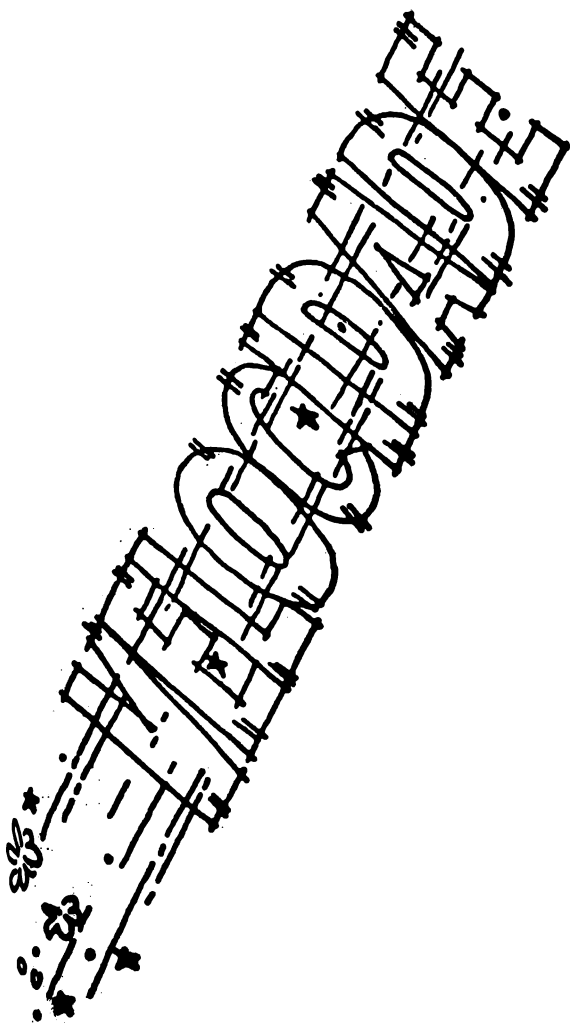
NOTAS

CAPÍTULO 18



MOVIMENTOS MAIS RÁPIDOS

18



MOVIMENTOS MAIS RÁPIDOS

Neste capítulo, mostraremos uma maneira de fazer gráficos que você vai gostar. Em muitos casos isso tornará a programação mais simples e aumentará a velocidade dos programas.

Digite:

```
PRINT CHR$(128) (ENTER)
```

O computador imprime um bloco preto igual a este:

Teste outros números. Digite:

```
PRINT CHR$(129) (ENTER)
PRINT CHR$(130) (ENTER)
PRINT CHR$(131) (ENTER)
```

O computador imprime três blocos com combinações diferentes de verde e preto:

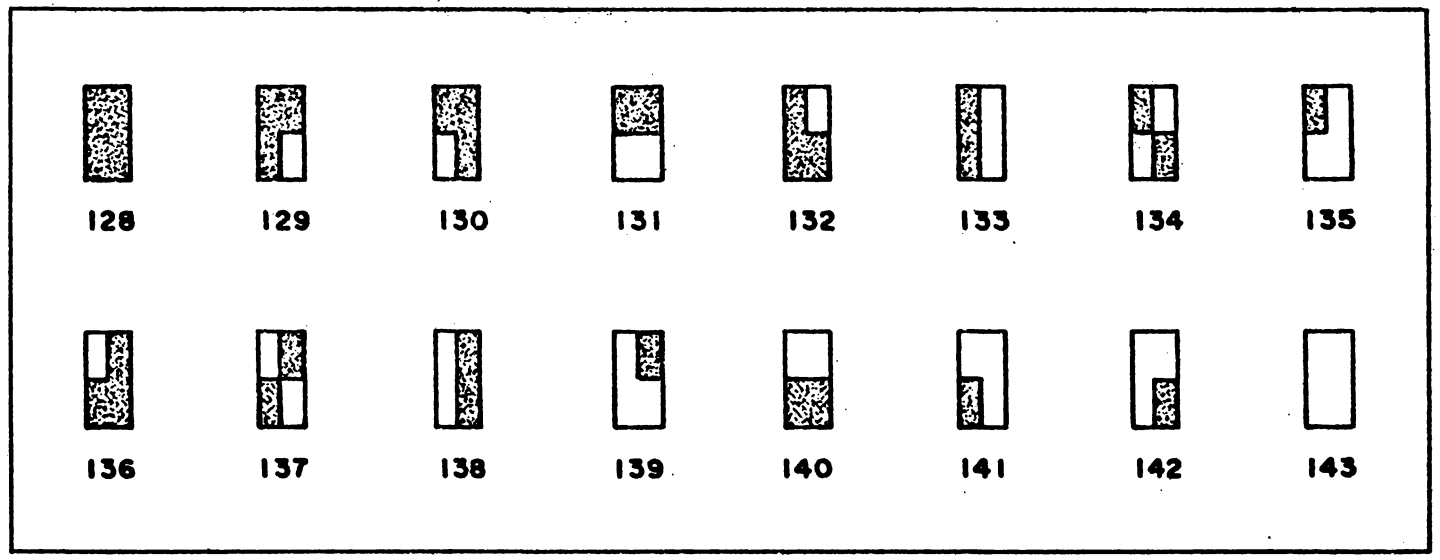
Como o fundo verde da tela dificulta a visualização do contorno dos blocos, digite este programa. Ele imprimirá o primeiro bloco com um fundo bege:

```
10 CLS(5)
20 PRINT@ 239, CHR$(129);
30 GOTO 30
```

Um gabarito das "Posições PRINT @ " é mostrado no apêndice C. No capítulo 7 mostramos como usá-lo. Certifique-se de que digitou o ponto e vírgula.

Lembra-se do CHR$ explicado no capítulo 15? CHR$ converte um código para o caráter correspondente. Por exemplo, CHR$(65) converte o código 65 para o caráter "A". Os códigos 128, 129, 130 e 131 são códigos de caracteres gráficos.

Veja no apêndice D, o gabarito de "Posicionamento de gráficos". Como explicamos anteriormente, as linhas mais escuras dividem o quadro em blocos. Estes blocos contêm quatro pontos e estes pontos podem ser arranjados em dezesseis maneiras diferentes para formar estes caracteres gráficos.



Você sabe por que é importante digitar um ponto e vírgula no fim das linhas PRINT @ ? Tente com e sem o ponto e vírgula.

O ponto e vírgula faz o computador parar de imprimir tão logo ele imprima o caráter. Sem o ponto e vírgula, ele continuaria imprimindo o fundo verde até o fim da linha.

Para imprimir todos os caracteres gráficos, digite e execute este programa:

```
10 CLS(5)
20 FOR C = 128 TO 143
30 PRINT @ 0,"PRESSIONE QUALQUER TECLA";
35 PRINT@ 32,"PARA CONTINUAR";
40 PRINT@ 173,C;
50 PRINT@ 240,CHR$(C) ;
60 K$=INKEY$ : IF K$="" THEN 60
70 NEXT C
80 GOTO 10
```

A linha 50 imprime os caracteres gráficos cujos códigos estão entre 128 e 143 na posição 240 da tela.

... Tente algo um pouco diferente. Digite:

```
PRINT CHR$(129+16) (ENTER)
```

O computador imprime o caráter gráfico de código 129, mas, a área que deveria estar verde, está amarela. Digite:

```
PRINT CHR$(129+32) (ENTER)
PRINT CHR$(129+48) (ENTER)
PRINT CHR$(129+64) (ENTER)
```

Estes são os números que você pode adicionar aos códigos de caracteres gráficos para criar cores diferentes:

Note que estes números são múltiplos de 16.

16 - amarelo
32 - azul
48 - vermelho
64 - bege
80 - azul esverdeado
96 - carmim
112 - laranja

Para ver todos os caracteres coloridos, acrescente estas linhas e execute o programa:

```
15 FOR X = 0 TO 7
17 IF X = 1 THEN CLS(1)
40 PRINT @ 170,C "+" X * 16;
50 PRINT @ 240,CHR$(C + X * 16);
75 NEXT X
```

Escreva três linhas para criar os caracteres abaixo.Faça o primeiro bege, o segundo carmim e o terceiro azul:

EXERCÍCIO DE PROGRAMAÇÃO

Respostas:

```
PRINT CHR$(133+64)
PRINT CHR$(137+96)
PRINT CHR$(140+32)
```

Como os caracteres gráficos são iguais as letras A, B, C ..., você pode tratá-los como

strings. É possível, por exemplo, combiná-los e armazená-los da mesma forma que seria feito com strings. Limpe a memória e digite:

```
10 A$=CHR$(129+32) + CHR$(131+32)
20 B$=CHR$(133+122) + CHR$(143+112) + CHR$(130+112)
```

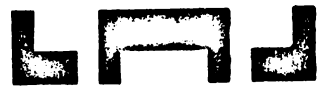
e você posicionará os caracteres gráficos no centro da tela da mesma forma que duas palavras (usando PRINT @). Digite:

Note a diferença. Você imprime caracteres gráficos usando PRINT @ no gabarito do apêndice C e acende pontos com SET usando o gabarito do apêndice D.

```
30 CLS(0)
40 PRINT @ 237,A$;
50 PRINT @ 241,B$;
60 GOTO 60
```

e execute o programa. O computador imprime a imagem de um carro azul e um caminhão laranja no centro da tela.

Escreva um programa que crie a imagem abaixo no centro da tela usando caracteres gráficos. Faça as cadeiras amarelas e a mesa laranja:



FAÇA VOCÊ MESMO O SEU PROGRAMA

Nós fizemos desta forma:

```
10 CL$=CHR$(139+16) + CHR$(130+16)
20 TA$=CHR$(142+112) + CHR$(140+112) + CHR$(141+112)
30 RC$=CHR$(129+16) + CHR$(135+16)
40 CLS(0)
50 PRINT@ 236,CL$+TA$+RC$;
60 GOTO 60
```

TRÂNSITO ENGARRAFADO

Limpe a memória e digite:

```
10 A=RND(7) * 16: B=RND(7) * 16
20 A$=CHR$(129+A) + CHR$(131+A)
30 B$=CHR$(133+B) + CHR$(143+B) + CHR$(130+B)
```

execute o programa e peça ao computador para imprimir A$ e B$. Execute e imprima A$ e B$, novamente. Repita várias vezes...

Toda vez que você executa o programa, o computador cria um carro e um caminhão randomicamente. Digite:

```
40 IF RND(2)=2 THEN VE$=A$ ELSE VE$=B$
```

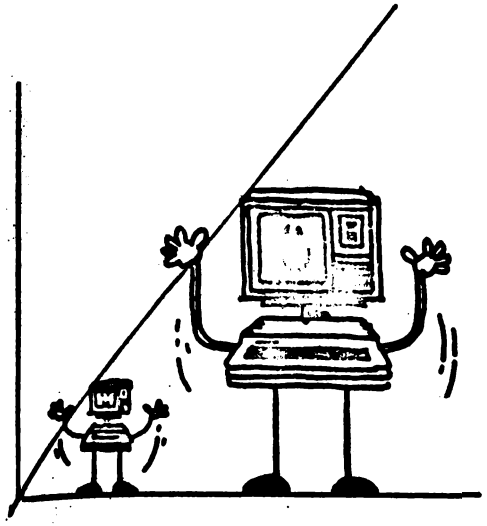
Execute o programa e imprima VE$ várias vezes. Algumas vezes você terá um carro e outras, um caminhão. O computador cria um carro ou um caminhão randomicamente.

Agora você pode engarrafar o trânsito. Digite:

Para rever o comando LEN consulte o capítulo 12.

```
50  IF LEN(TR$+VE$+SP$) > 32 THEN 100
60  TR$ = TR$ + VE$
70  GOTO 10
100 PRINT TR$
```

Execute o programa várias vezes. O computador sempre cria um engarrafamento de trânsito com 32 caracteres de comprimento. Para fazê-lo movimentar-se, digite:



```
100 INPUT "VELOCIDADE (1-200)"; S
110 FOR P = 0 TO 480
120 PRINT@ P, TR$;
130 FOR X = 1 TO S : NEXT X
140 CLS(0)
150 NEXT P
```

e execute. O trânsito vai se mover do canto superior esquerdo até o canto inferior direito da tela.

A linha 120 imprime o trânsito na posição P (0 até 480).

A linha 130 faz uma pausa no programa cuja duração depende da velocidade informada.

A linha 140 limpa a tela para que o trânsito possa ser impresso na próxima posição.

Para fazer o trânsito mover-se na tela sem parar, digite:

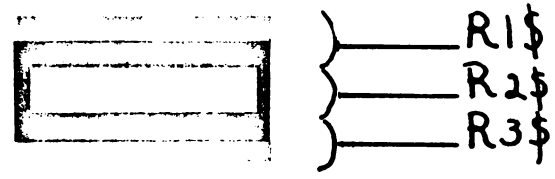
No capítulo 12 mostramos como usar os comandos LEFT$ e RIGHT$.

```
110 P = 320
150 P = P + 1
160 IF P = 351 THEN 110
170 PRINT@ P, LEFT$(TR$,352-P);
180 PRINT@ 320, RIGHT$(TR$,P-320);
190 GOTO 130
```

e execute.

FAZENDO GRANDES FIGURAS

Até agora, neste capítulo, não acrescentamos altura nos gráficos. Para imprimir a boca a-baixo, precisaremos de três linhas de string:



Para criar R1$ e R3$, limpe a memória e digite:

```
10 FOR X = 1 TO 9
20 R1$=R1$ + CHR$(131+48)
30 R3$=R3$ + CHR$(140+48)
40 NEXT X
```

Execute o programa e imprima R1$ e R3$. R1$ agora contém um string de nove caracteres gráficos de número 131+48 e R3$ contém nove caracteres de número 140+48.

Para criar R2$, digite:

```
50 T1$=CHR$(137+64)
60 T2$=CHR$(136+64)
70 R2$=CHR$(138+48)+T1$+T1$+T2$+T1$+T1$+T2$+T1$+CHR$(133+48)
```

e para imprimir a boca interior na tela, digite:

```
80  CLS
90  PRINT@5, "POSICAO";
100 INPUT L
110 CLS(0)
120 PRINT @ L, R1$;
130 PRINT @ L+32, R2$;
140 PRINT @ 2+64, R3$;
150 GOTO 90
```

A linha 120 imprime a primeira linha — R1$ — na posição L que você forneceu. Vamos assumir a posição 40.

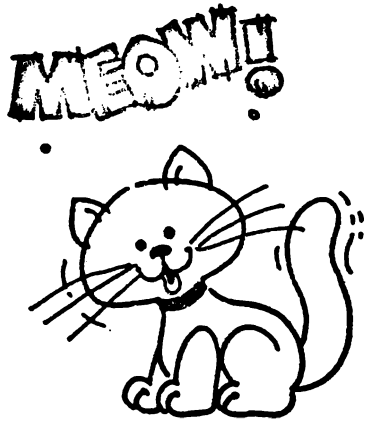
A linha 130 imprime a segunda linha — R2$ — na posição L+32 que é a posição 72. Note que existem 32 posições em uma fila (0-31) e a posição L+32 está diretamente abaixo da posição L.

A linha 140 imprime R3$ na posição L+64, que está diretamente abaixo da posição L+32 (posição de R2$).

Com este método, fizemos, para cada linha, um string, mas, poderíamos desenhar toda a boca num único string chamado MO$. Para fazer isto, precisamos combinar todas as linhas — R1$, R2$ e R3$ — mais BK$ (o fundo entre as linhas).

Digite:

```
72 FOR X = 1 TO 32-9
74 BK$ = CHR$(128) + BK$
76 NEXT X
78 MO$=R1$+BK$+R2$+BK$+R3$
```



Já que toda a boca é agora um string único, você só precisa de uma instrução PRINT @ . Elimine as linhas 130 e 140 e mude a linha 120:

```
130
140
120 PRINT@L, MO$;
```

e execute-o.

Criando strings com caracteres gráficos, você pode fazer com que os programas animados sejam executados mais rapidamente. No próximo capítulo mostraremos como fazer um computador dançarino com estes strings gráficos.

ASSUNTOS APRENDIDOS NO CAPÍTULO 18

CONCEITO BASIC

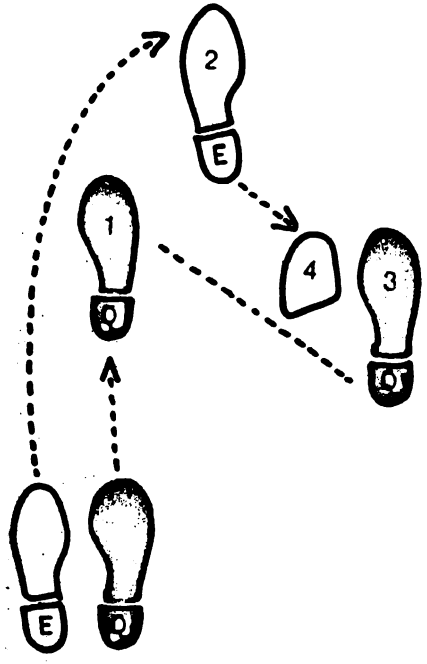
Caracteres gráficos

CAPÍTULO 19

VAMOS DANÇAR

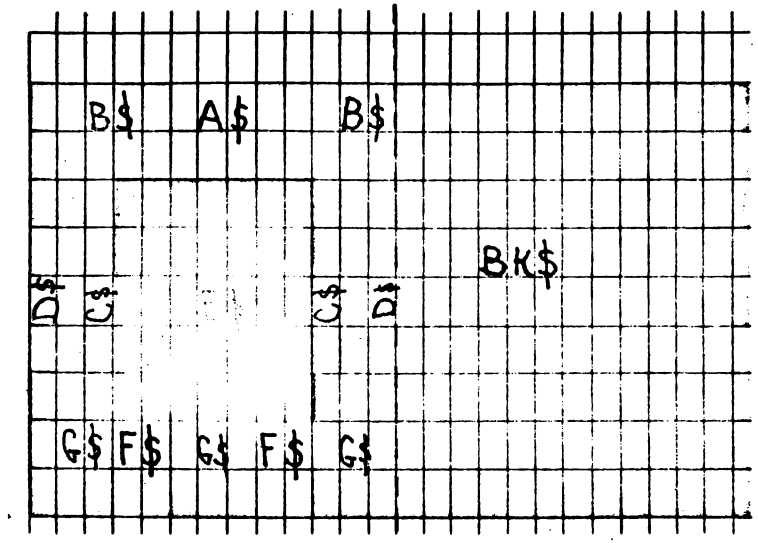
19

VAMOS DANÇAR



Este capítulo lhe dará a oportunidade de rever o que você já aprendeu. Não ensinaremos nada de novo, apenas nos divertiremos um pouco fazendo o computador dançar. Para isto, vamos usar "strings gráficos".

Aqui ele está parado:



Como isto ocupará uma área de string muito grande, digite:

```
1 CLEAR 1000
```

para reservar bastante espaço.

Para criar string com caracteres pretos, digite:

```
10 D$ = CHR$(128) + CHR$(128)
20 G$ = D$ + CHR$(128)
30 B$ = G$ + D$
40 BK$ = B$ + B$ + B$ + D$ + D$
```

Na tela, o verde claro aparecerá bege, o verde escuro, vermelho e a área cinza será preta.

execute o programa e faça o computador imprimir B$, D$, G$ e BK$.

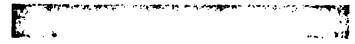
Na verdade B$ tem um comprimento de 5 caracteres. Na tela, ele aparecerá alinhado com a palavra PRINT.

```
PRINT B$ (ENTER)
 
PRINT D$ (ENTER)

PRINT G$ (ENTER)
```

As outras variáveis tem os seguintes comprimentos:

D$ - 2 caracteres
G$ - 3 caracteres
BK$ - 19 caracteres
A$ - 3 caracteres

PRINT BK$ (ENTER)



Para criar os strings de cor bege, digite:

```
50 C$ = CHR$(143+64)
60 F$ = C$ + C$
70 A$ = F$ + C$
```

Estas variáveis possuem os seguintes comprimentos:

C$ - 1 caráter
F$ - 2 caracteres
E$ - 7 caracteres

execute o programa e imprima as variáveis A$, C$ e F$:

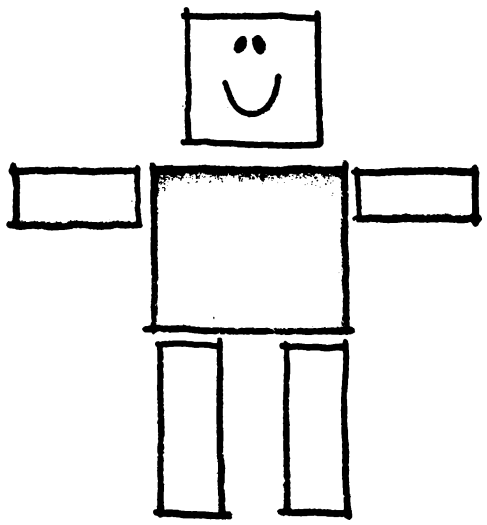
```
PRINT A$ (ENTER)
PRINT C$ (ENTER)
PRINT F$ (ENTER)
```

Para formar o string de cor vermelha (E$) digite:

```
80  FOR X = 1 TO 7
90  E$ = E$ + CHR$(143+48)
100 NEXT X
```

execute o programa e imprima E$:

PRINT E$ (ENTER)



Crie os string HD$, BD$ e L1$, de forma que depois de executar o programa, você possa imprimi-los desta maneira:

PRINT HD$ (ENTER)

PRINT BD$ (ENTER)

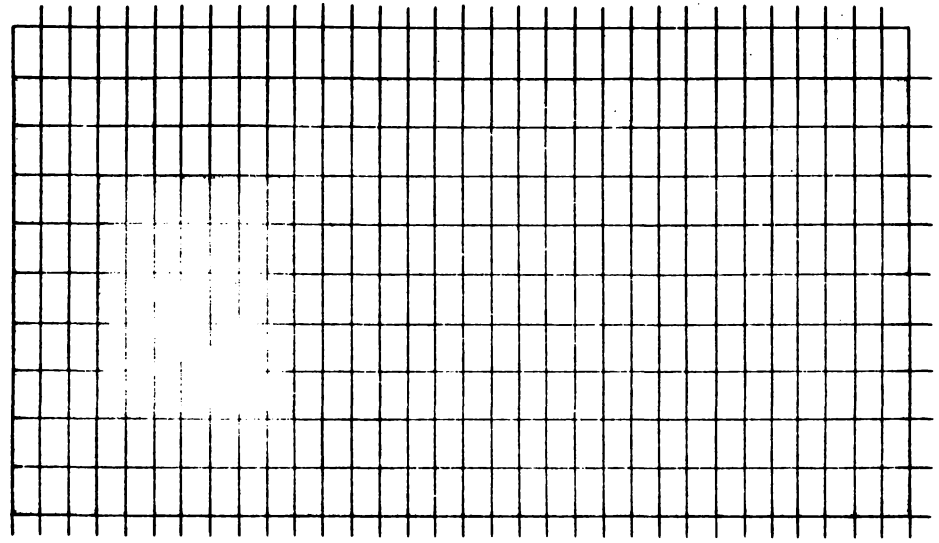
PRINT L1$ (ENTER)

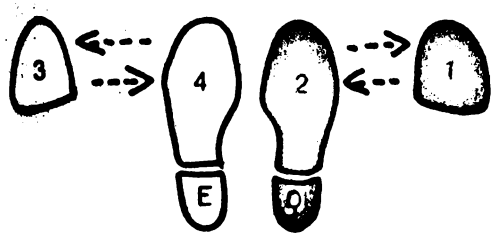
EXERCÍCIO DE PROGRAMAÇÃO

Aqui está como fizemos:

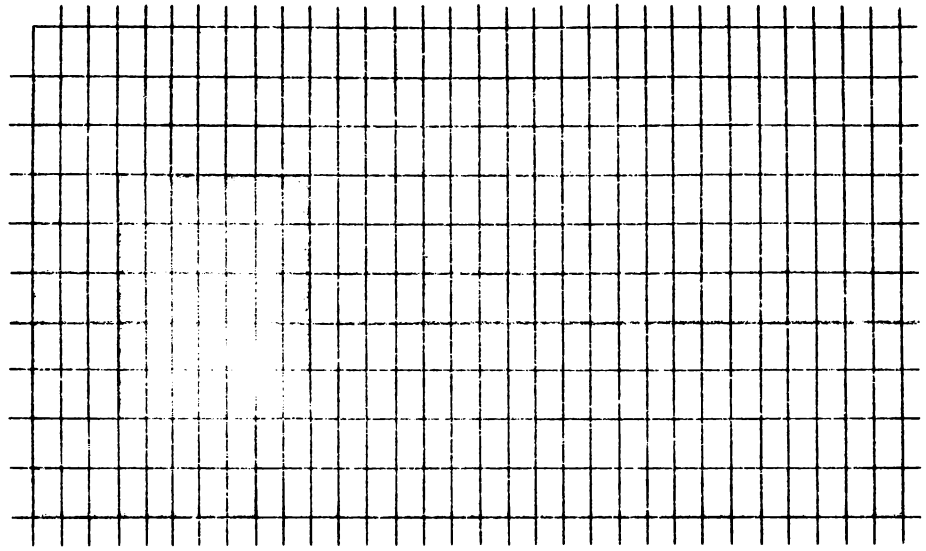
```
110 HD$ = B$+A$+B$+BK$+B$+A$+B$+BK$
120 FOR X = 1 TO 4
130 BD$ = BD$+D$+C$+E$+C$+D$+BK$
140 NEXT X
150 L1$ = G$+E$+G$+BK$+G$+F$+G$+F$+G$+BK$+G$+F$+G$+F$+G$
```

Para fazer o computador dançar, temos que fornecer-lhe mais duas posições para as pernas:





Acrescente algumas linhas ao programa para criar os strings L2$ e L3$.



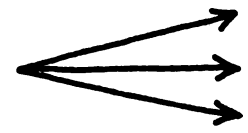
EXERCÍCIO DE PROGRAMAÇÃO

Eis a nossa solução:

```
160 H$  = G$+G$
170 I$  = H$+D$
180 L2$ = G$+E$+A$+BK$+G$+F$+H$+F$+BK$+G$+F$
190 L3$ = A$+E$+G$+BK$+F$+H$+F$+G$+BK$+I$+F$
```

Para ver as três posições do computador, acrescente estas linhas ao programa:

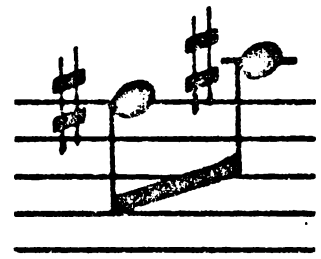
```
500  INPUT "LOCALIZACAO (0-243)"; L
510  INPUT "POSICAO (1-3)"; P
520  GOSUB 1000
530  GOTO 500

1000 CLS(0)
1010 PRINT@L, HD$ + BD$;
1020 ON P GOSUB 2000, 3000, 4000
1030 PRINT@L+32*6, LG$;: RETURN

2000 LG$ = L1$: RETURN
3000 LG$ = L2$: RETURN
4000 LG$ = L3$: RETURN
```



execute o programa. Experimente outras localizações e outras posições.

A linha 1010 imprime a cabeça e o corpo na localização escolhida.

A linha 1020 desvia o programa para uma subrotina que iguala LG$ a L1$, L2$ ou L3$ (dependendo do valor fornecido para a variável P na linha 510).



Consulte o capítulo 10 para rever as instruções DATA, READ e RESTORE.

A linha 1030 imprime então LG$ imediatamente abaixo da cabeça e do corpo, ficando 6 linhas abaixo da localização escolhida anteriormente.

Através do controle destas localizações e posições, você pode fazer o computador dançar. Aqui está como fizemos. Altere as linhas 500 e 510 e acrescente estas outras:

```
500  FOR X = 1 TO 17
510  IF X=1  OR X=5 THEN RESTORE

5    INPUT "VELOCIDADE (1-10)"; V
515  READ L, P, T, D
525  SOUND T, V*D
527  NEXT X

5000 DATA 137, 2,  89, 1, 240, 1, 133,2
5010 DATA 137, 3, 159, 1, 229, 1, 133,2
5020 DATA   5, 1,  89, 1, 229, 1, 133,2
5030 DATA   5, 1, 147, 1, 229, 1, 159,1
5040 DATA 229, 1, 147, 1,   5, 1, 133,1
5050 DATA 229, 1, 125, 2,   5, 1, 133,1
5060 DATA 229, 1, 147, 2
```

execute o programa e veja-o dançar.

A linha 515 lê a localização, a posição, o tom e sua duração que estão nas linhas 5000 a 5060. Na primeira passagem pelo programa, o computador aparecerá na localização 137 e po-

sição 2. Na linha 525, o computador fará soar o tom número 89 com uma duração V*1. Na segunda passagem pelo programa, o computador aparecerá na localização 240 e posição 1 soando o tom 133 com uma duração V*2.

Como você pode ver, para tornar isto mais divertido, basta acrescentar mais posições. Tente fazê-lo ou então, dê uma olhada nos nossos exemplos no fim deste manual para ter algumas idéias a respeito do uso de caracteres gráficos.

SEÇÃO III

APLICAÇÕES COMERCIAIS

Você por acaso tem alguma informação que gostaria que o computador manipulasse? Nesta seção faremos o computador ordenar, comparar, armazenar e imprimir seus dados mais rápido e mais precisamente do que você conseguiria com suas próprias mãos.

Aqui está uma lista de coisas com as quais, já é sabido que o computador pode lidar:

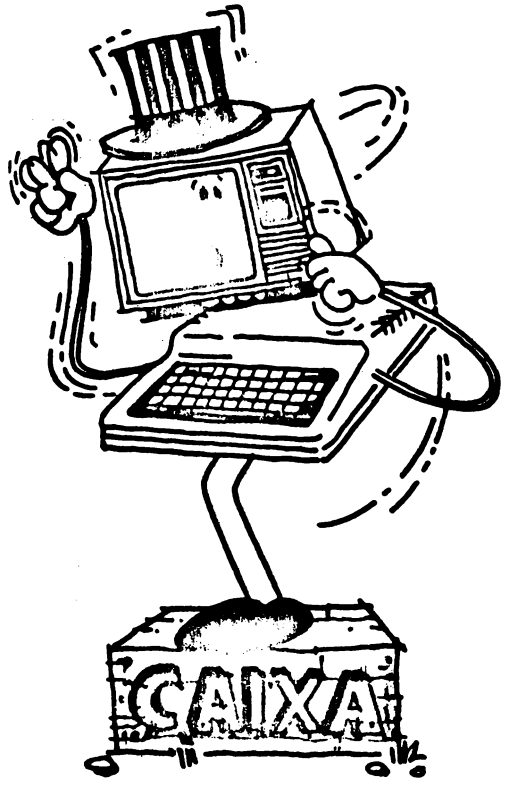
1 - lista de compras
2 - controle de talão de cheques
3 - contas a pagar
4 - controle de despesas médicas
5 - lista de endereços
6 - lista de telefone
7 - anotações
8 - controle de estoque
9 - coleção de livros
10 - coleção de moedas
11 - coleção de selos
12 - coleção de discos
13 - relação de programas
14 - controle de vendas
15 - contas a receber
16 - cartas
17 - poemas
18 - músicas

CAPÍTULO 20

CONTROLANDO TUDO!

20

CONTROLANDO TUDO



Você já tentou escrever um programa que lidasse com muitas informações? Caso já tenha tentado, você ficará feliz em saber que o computador possui um sistema de "organização de índices" que facilita este tipo de programação.

Para começar, como faria você para armazenar no computador estes resultados de uma eleição?

R E S U L T A D O S D A E L E I Ç Ã O

MUNICÍPIO	VOTOS APURADOS PARA O CANDIDATO A
1	143
2	215
3	125
4	331
5	442
6	324
7	213
8	115
9	318
10	314
11	223
12	152
13	314
14	92

Para armazenar dados, nós sempre utilizamos variáveis. Para fazer com que o computador armazene os votos dos 3 primeiros municípios, digite:

Consulte o capítulo 2 para uma rápida revisão do conceito de variável.

```
A = 143 (ENTER)
B = 215 (ENTER)
C = 125 (ENTER)
```

... mas, existe uma maneira muito melhor. Digite o seguinte:

```
A(1) = 143 (ENTER)
A(2) = 215 (ENTER)
A(3) = 125 (ENTER)
```

Cada uma destas variáveis possui um índice - A(1), A(2) e A(3). Apesar do índice, estas variáveis são exatamente iguais às anteriores. Para verificar que elas funcionam da mesma forma, digite as linhas abaixo:

As pessoas que lidam com computadores chamam de VETOR, a lista inteira de variáveis indexadas. Cada variável indexada é um item do vetor.

```
PRINT A; B; C; (ENTER)
PRINT A(1); A(2); A(3) (ENTER)
```

Ambas funcionam da mesma forma,certo? Então, por que variáveis indexadas são melhores? Dê uma olhada nestes dois programas,ambos fazem a mesma coisa:

PROGRAMA 1

```
10  DATA 143, 215, 125, 331, 442
```

```
20  DATA 324, 213, 115, 318, 314
30  DATA 223, 115, 314,  92
40  READ A, B, C, D, E
50  READ F, G, H, I, J
60  READ K, L, M, N
70  INPUT "MUNICIPIO NO. (1-14)"; Z
75  IF Z>14 THEN 70
80  IF Z= 1 THEN PRINT A "VOTOS"
90  IF Z= 2 THEN PRINT B "VOTOS"
100 IF Z= 3 THEN PRINT C "VOTOS"
110 IF Z= 4 THEN PRINT D "VOTOS"
120 IF Z= 5 THEN PRINT E "VOTOS"
130 IF Z= 6 THEN PRINT F "VOTOS"
140 IF Z= 7 THEN PRINT G "VOTOS"
150 IF Z= 8 THEN PRINT H "VOTOS"
160 IF Z= 9 THEN PRINT I "VOTOS"
170 IF Z=10 THEN PRINT J "VOTOS"
180 IF Z=11 THEN PRINT K "VOTOS"
190 IF Z=12 THEN PRINT L "VOTOS"
200 IF Z=13 THEN PRINT M "VOTOS"
210 IF Z=14 THEN PRINT N "VOTOS"
220 GOTO 70
```

PROGRAMA 2

```
10  DATA 143, 215, 125, 331, 442
```

```
20  DATA 324, 213, 115, 318, 314
30  DATA 223, 152, 314,  92
40  DIM A(14)
50  FOR X=1 TO 14
60  READ A(X)
70  NEXT X
80  INPUT "MUNICIPIO NO. (1-14)"; Z
85  IF Z>14 THEN 80
90  PRINT A(Z) "VOTOS"
100 GOTO 80
```

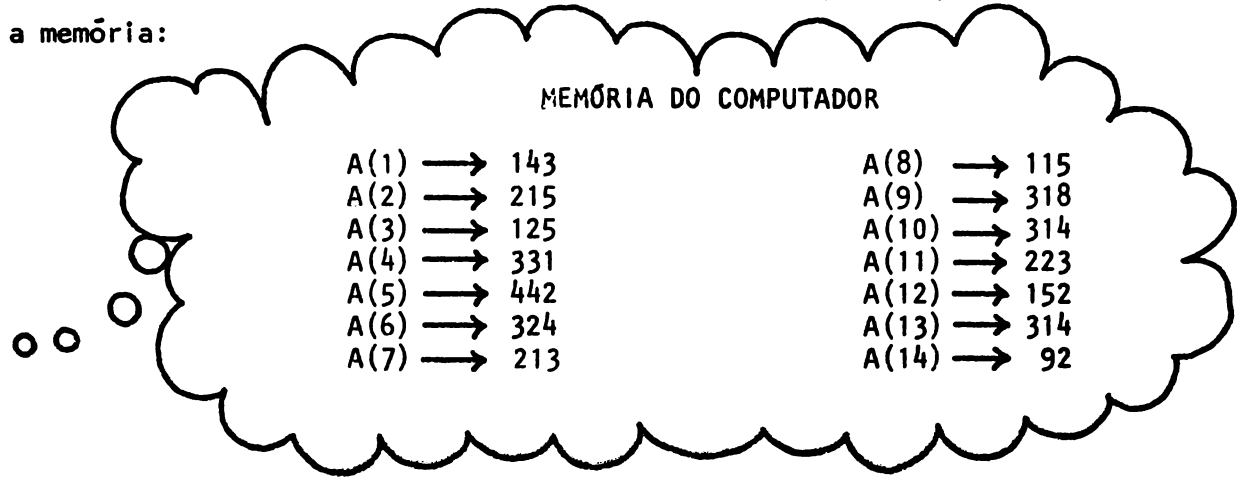
O primeiro programa usa variáveis normais enquanto que o segundo usa variáveis indexadas. As variáveis indexadas simplificam muito a programação de grandes listas de informações.

Digite e execute o segundo programa. Aqui está como ele funciona:

Na verdade, é reservada uma área para 15 itens, se for levado em conta o índice 0.

A linha 40 reserva espaço para uma lista - um vetor - de informações chamada A, com 14 itens indexados.

As linhas 50 e 70 estabelecem um loop de 1 a 14 enquanto que a linha 60 lê os dados para a memória:



Isto armazena todos os votos em um vetor chamado A, que contém 14 itens indexados.

A linha 80 pede que você forneça um índice, a linha 90 imprime o número de votos no município indicado.

Agora que os votos estão armazenados num vetor, fica fácil trabalhar com eles. Por exemplo, se você quiser que o programa seja capaz de modificar um dos totais de votos, acrescente as seguintes linhas:

```
92 INPUT "VOCE QUER ACRESCENTA ALGUM VOTO"; R$
94 IF R$="NAO" GOTO 80
96 INPUT "QUANTOS VOTOS MAIS?"; X
97 A(Z)=A(Z)+X
98 PRINT "O TOTAL DE VOTOS NO MUNICIPIO" Z "E' AGORA" A(Z)
```

O nome deste vetor é A. O 'X' ou 'Z' entre parênteses indica o índice de um dos itens da lista.

Ou se você quiser imprimir uma tabela como a do início deste capítulo, você pode acrescentar estas linhas:

```
72  INPUT S$
74  IF S$="SIM" THEN GOSUB 110
110 PRINT "MUNICIPIO","VOTOS"
120 FOR X=1 TO 14
130 PRINT X, A(X)
140 NEXT X
150 RETURN
```

e modifique a linha 100:

```
100 GOTO 72
```

Você não precisa estudar este programa se estiver ancioso para seguir em frente. Nós só estamos mostrando algumas das vantagens do uso de variáveis indexadas.

RESERVANDO ESPAÇO PARA O CANDIDATO B

Suponhamos que você também quer acompanhar os votos do candidato B.

RESULTADO DAS ELEIÇÕES

MUNICÍPIO	VOTOS PARA O CANDIDATO A	VOTOS PARA O CANDIDATO B
1	143	678
2	215	514
3	125	430
4	331	475
5	442	302
6	324	520
7	213	613
8	115	694
9	318	420
10	314	518
11	223	370
12	152	412
13	314	460
14	92	502

Nós podemos simplesmente acrescentar um outro vetor para armazenar os dados do candidato B.

Vamos chamar este vetor de B. Este programa armazena os votos dos candidatos A (vetor A) e B (vetor B):

```
10  DATA 143, 215, 125, 331, 442
20  DATA 324, 213, 114, 318, 314
30  DATA 223, 152, 314,  92
40  DATA 678, 514, 430, 475, 302
50  DATA 520, 613, 694, 420, 518
60  DATA 379, 412, 460, 502
70  DIM A(14), B(14)
80  FOR X=1 TO 14
90  READ A(X)
100 NEXT X
110 FOR X=1 TO 14
120 READ B(X)
130 NEXT X
140 INPUT "MUNICIPIO NO."; Z
145 IF Z>14 THEN 140
150 INPUT "CANDIDATO A OU B"; R$
160 IF R$="A" THEN PRINT A(Z)
170 IF R$="B" THEN PRINT B(Z)
180 GOTO 140
```

dados para o vetor A

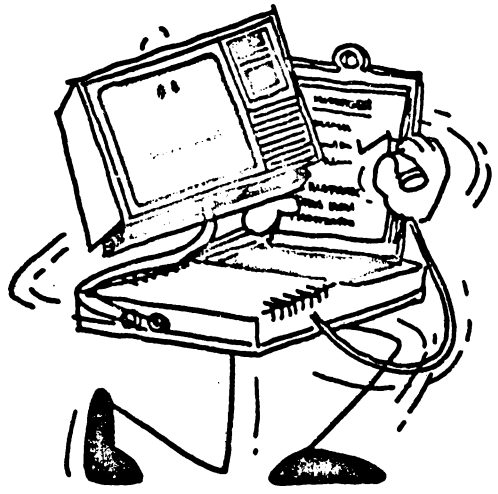
dados para o vetor B

reserva de área

leitura dos dados do vetor A

leitura dos dados do vetor B

CONTROLE DE ESTOQUE



Frequentemente usamos variáveis indexadas para controlar dados de uso comercial. Escreva um programa para ajudar uma loja a controlar o seu estoque de 12 itens:

E S T O Q U E

ITEM Nº	QUANTIDADE
1	33
2	12
3	42
4	13
5	15
6	23
7	25
8	30
9	33
10	27
11	14
12	8

FAÇA VOCÊ MESMO O SEU PROGRAMA

Aqui está o programa que escrevemos:

```
10 DATA 33, 12, 42, 13, 15, 23
20 DATA 25, 30, 33, 27, 14,  8
30 DIM E(12)
40 FOR X=1 TO 12
50 READ E(X).
60 NEXT X
70 INPUT "ITEM NO."; N
75 IF N>12 THEN 70
80 PRINT "O ESTOQUE DO ITEM" N "E''' E(N)
90 GOTO 70
```

TESTE DE MEMÓRIA

Na verdade, você não precisa usar o comando DIM se nenhum dos seus vetores possui mais de 10 itens. Entretanto, é sempre aconselhável colocar este comando para reservar o tamanho exato de área de memória.

Lembre-se, você pode colocar várias instruções numa mesma linha separando-as com dois pontos. Estas instruções fazem uma pausa de 10 segundos no programa.

O computador usa uma variável indexada para memorizar estes números, e você, o que está usando?

Respire fundo e prepare-se; este programa testa a sua memória e a do computador. Digite NEW para apagar o programa anterior e digite:

```
5   DIM A(7)
10  PRINT "DECORE ESTES NUMEROS"
15  PRINT "VOCE TEM 10 SEGUNDOS"
20  FOR X=1 TO 7
30  A(X)=RND(100)
40  PRINT A(X)
50  NEXT X
60  FOR X=1 TO 460*10: NEXT X
70  CLS
80  FOR X=1 TO 7
90  PRINT "QUAL ERA O NUMERO" X
100 INPUT R
110 IF A(X)=R THEN PRINT "CERTO" ELSE PRINT "ERRADO - ERA" A(X)
120 NEXT X
```

A linha 5 reserva área para uma variável indexada de nome A com 7 itens.

As linhas 20 a 50 armazenam 7 números aleatórios na variável indexada A.

A linha 60 introduz uma pausa de 10 segundos no programa, e a linha 70 limpa a tela.

As linhas 80 a 120 verificam se você ainda se lembra dos números que estão armazenados em A.

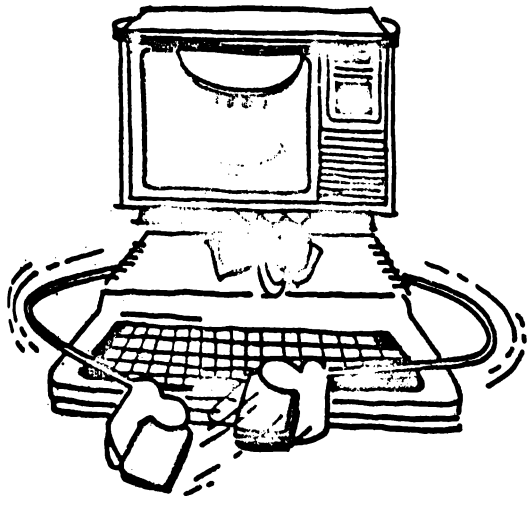
DANDO AS CARTAS

Preste atenção, agora vamos ensinar-lhe a usar o computador para dar cartas.

Para dar 52 cartas aleatoriamente, apague o programa anterior e digite:

```
40  FOR X=1 TO 52
50  C=RND(52)
90  PRINT C;
100 NEXT X
```

Se você quiser, tente fazer isto sozinho. Neste caso, devemos preveni-lo, não é muito fácil.



execute o programa... O computador dá 52 cartas aleatoriamente, mas, entretanto, se você observar com cuidado, verá que algumas das cartas são iguais.

Temos, então, que achar uma maneira de controlar quais as cartas que já foram dadas. Para tanto, vamos primeiro criar uma variável indexada chamada T que conterá todas as 52 cartas. Digite:

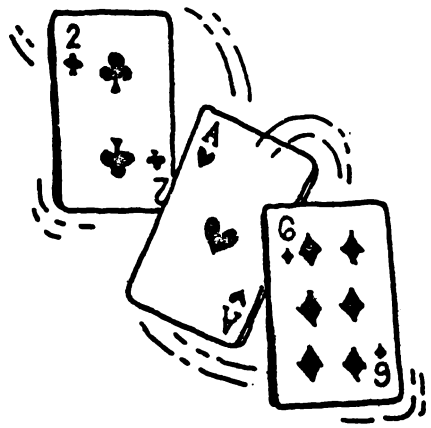
```
5  DIM T(52)
10 FOR X=1 TO 52
20 T(X)=X
30 NEXT X
```

Isto simplesmente diz ao computador que T(1)=1, T(2)=2, T(3)=3, e assim por diante até T(52) que obviamente é igual a 52.

Agora acrescente estas linhas, que anularão no vetor T, cada carta dada. Digite:

```
60 IF T(C)=0 THEN 50
80 T(C)=0
```

Agora, o computador não pode mais dar duas cartas iguais. Por exemplo, suponhamos que a primeira carta dada seja um dois. A linha 80 modifica o valor de T(2) para zero. Agora suponhamos que o computador dê outra carta número dois, neste caso, a linha 60 obriga-o a retornar à linha 50 e fornecer uma outra carta, porque T(2) vale zero.



Execute o programa e repare como o computador hesita antes de imprimir as últimas cartas. Isto ocorre porque ele está testando várias cartas "aleatórias" antes de achar uma que ainda não tenha sido dada.

Se quiser jogar cartas com o computador, você tem que fazer com que ele memorize as cartas que já foram dadas. Para tanto, podemos criar uma outra variável indexada que chamaremos de D. Digite:

```
7  DIM D(52)
70 D(X)=T(C)
90 PRINT D(X);
```

Agora a variável indexada D contém uma lista de todas as cartas que o computador deu, na

Este programa é um pouco demorado. Se ele estiver atrasando-o, pule-o e retorne a ele mais tarde.

ordem que elas foram dadas.

Como você modificaria este programa, de forma que ele imprima apenas a sua "mão" - as cinco primeiras cartas dadas?

FAÇA VOCÊ MESMO O SEU PROGRAMA

Aqui está a nossa resposta:

```
5   DIM T(52)
7   DIM D(52)
10  FOR X=1 TO 52
20  T(X)=X
30  NEXT X
34  CLS
36  PRINT @ 101, "... DANDO AS CARTAS"
```

A linha 130 indica que a próxima impressão ocorrerá na posição 167.

```
40  FOR X=1 TO 52
50  C=RND(52)
60  IF T(C)=0 THEN 50
70  D(X)=C
75  SOUND 128,1
80  T(C)=0
100 NEXT X
110 CLS
120 PRINT @ 107,"SUA MAO"
130 PRINT @ 167,
140 FOR X=1 TO 5
150 PRINT D(X);
160 NEXT X
```

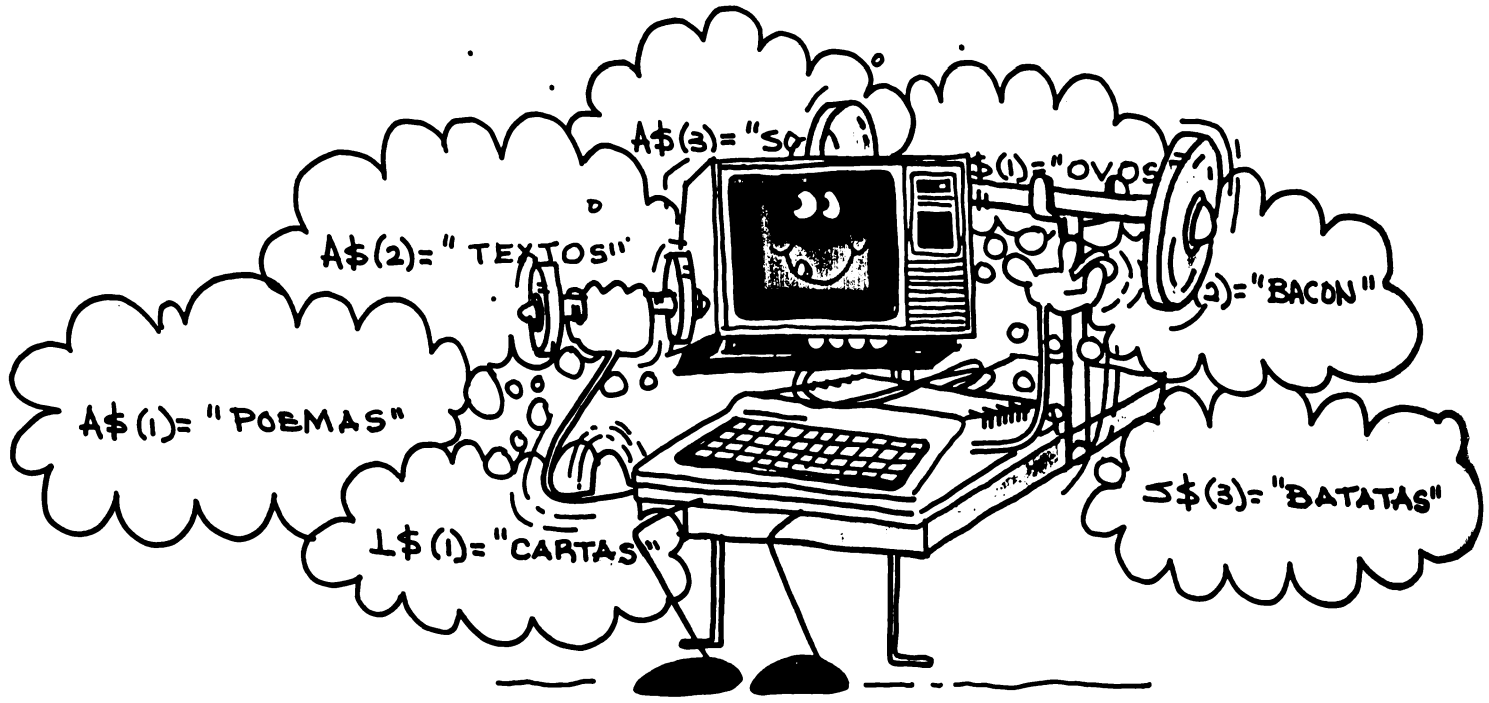
Vamos lhe mostrar, nos próximos capítulos, muito mais coisas que você pode fazer com variáveis indexadas. Isto é tudo que você tem que se lembrar:

REGRAS DE VARIÁVEIS INDEXADAS

1. Existem dois tipos de variáveis:
   A. VARIÁVEIS SIMPLES, tais como A, B, C e D
   B. VARIÁVEIS INDEXADAS ou VETORES, tais como A(5),A(3),B(2) e B(6)
2. Um VETOR é um grupo de itens indexados que pertencem todos a uma variável de mesmo nome. Por exemplo, M(2),M(4),M(5) e M(6) pertencem todos ao vetor de nome M.

ASSUNTOS APRENDIDOS NO CAPÍTULO 20	
PALAVRA BASIC	CONCEITO BASIC
DIM	Variáveis indexadas

CAPÍTULO 21



REFORÇANDO A SUA ESCRITA

21

REFORÇANDO A SUA ESCRITA

No capítulo anterior, usamos apenas vetores de números, mas, eles também podem ser usados com palavras. Você verá neste capítulo que o computador é capaz de armazenar, editar e imprimir um texto cheio de palavras.

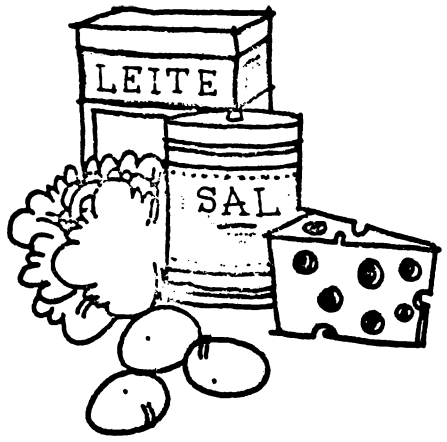
Vamos começar com uma simples lista de palavras — uma lista de compras:

1. OVOS	4. SAL	7. TOMATES	10. PEIXE
2. BACON	5. AÇÚCAR	8. LEITE	11. LARANJA
3. BATATAS	6. ALFACE	9. QUEIJO	12. ABACAXI

Para fazer com que o computador se lembre destes itens, vamos guardar cada um numa variável string indexada. Por exemplo, para os três primeiros itens, digitaremos:

Veja o símbolo $, é a única diferença entre estas variáveis indexadas e as que foram usadas no capítulo anterior.

```
S1$ = "OVOS"   (ENTER)
S2$ = "BACON"  (ENTER)
S3$ = "BATATAS" (ENTER)
```



Para fazer com que o computador imprima estes três primeiros itens, digite:

PRINT S1$, S2$, S3$ (ENTER)

Aqui está como colocá-los dentro de um programa:

```
5   DIM S$(12)
10  DATA OVOS, BACON, BATATAS, SAL
20  DATA ACUCAR, ALFACE, TOMATES, LEITE
30  DATA QUEIJO, PEIXE, LARANJA, ABACAXI
40  FOR X=1 TO 12
50  READ S$(X)
60  NEXT X
70  PRINT "LISTA DE COMPRAS"
80  FOR X=1 TO 12
90  PRINT X; S$(X)
100 NEXT X
```

leitura dos itens para dentro de S$

impressão do vetor S$

Este programa armazena os 12 itens num vetor chamado S$ e imprime a lista. Acrescente algumas linhas de forma que seja possível modificar qualquer um dos itens.

FAÇA VOCÊ MESMO O SEU PROGRAMA

Aqui estão as linhas que acrescentamos:

```
110 INPUT "QUAL O NO. DO ITEM QUE VOCE QUER MODIFICAR"; N
115 IF N>12 THEN 110
120 INPUT "QUAL E' O NOVO ITEM"; S$(N)
130 GOTO 80
```

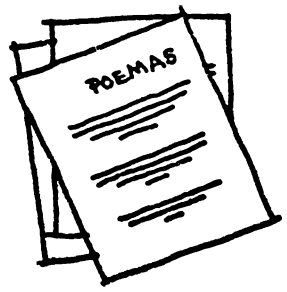
No fim deste manual, mostraremos como acrescentar e retirar itens desta lista.

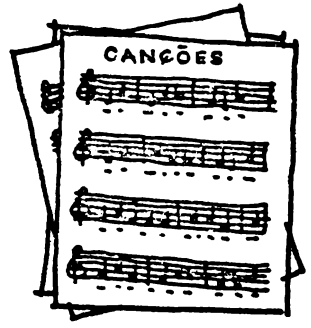
Você quer compor uma música? Dê uma olhada no "compositor de músicas" no apêndice de programas simples deste manual.

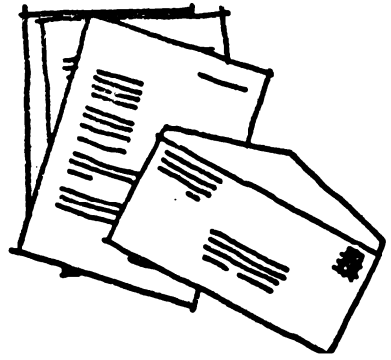
ESCREVENDO LETRAS DE MÚSICA

(...UM POEMA, UMA CARTA,...)

Aqui está um programa que utiliza um vetor para ajudá-lo a escrever letras de música. Limpe a memória e digite:







```
5  DIM A$(4)
10 PRINT "DIGITE 4 LINHAS"
20 FOR X=1 TO 4
30 INPUT A$(X)
40 NEXT X
50 CLS
60 PRINT "ESTA E' A SUA CANCAO"
70 PRINT
80 FOR X=1 TO 4
90 PRINT X; " "; A$(X)
95 NEXT X
```

leitura dos dados para dentro de A$

impressão do vetor A$

Execute-o e acrescente algumas linhas para poder rever as linhas da música.

FAÇA VOCÊ MESMO O SEU PROGRAMA

Aqui estão as linhas que acrescentamos:

```
110 PRINT
120 INPUT "QUE LINHA VOCE QUER REVER"; L
125 IF L>4 THEN 120
130 PRINT "DIGITE A NOVA LINHA"
140 INPUT A$(L)
150 GOTO 50
```

Você já ouviu falar em processador de palavras? É um programa que permite criar um texto no computador, fazer alterações nele e imprimí-lo.

ESCREVENDO UM TEXTO

(...UM ROMANCE, UM ARTIGO...)

Aqui está um programa para ajudá-lo a escrever textos. Use-o com o que foi ensinado no capítulo 12 e você terá um processador de palavras:

Você não se lembra mais de alguns comandos usados neste programa? O comando CLEAR foi explicado no capítulo 12 e INKEY$ foi abordado no capítulo 13.

```
1   CLEAR 1000
5   DIM A$(50)
10  PRINT "ESCREVA UM PARAGRAFO"
20  PRINT "PRESSIONE </> QUANDO TERMINAR"
30  X=1
40  A$=INKEY$
50  IF A$="" THEN 40
60  PRINT A$;
```


```
70  IF A$="/" THEN 110
80  A$(X)=A$(X) + A$
90  IF A$="." THEN X=X+1
100 GOTO 40
110 CLS
120 PRINT "SEU PARAGRAFO:"
130 PRINT
140 FOR Y=1 TO X
150 PRINT A$(Y);
160 NEXT Y
```

Digite e execute o programa e antes de ler como funciona, faça experiências com ele. Digite:

```
PRINT A$(1) (ENTER)
PRINT A$(2) (ENTER)
PRINT A$(3) (ENTER)
```

Deu para perceber como funciona? Aqui está a explicação:

A linha 1 reserva 1000 bytes para variáveis strings.

A linha 5 reserva espaço para um vetor chamado A$ que pode conter até 50 sentenças.

A linha 30 faz X igual a 1. Esta variável será usada como índice das sentenças.

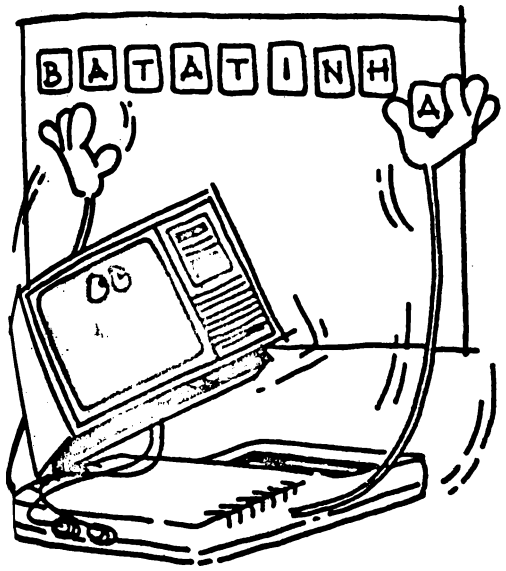
A linha 40 testa a tecla que está sendo pressionada. Se for nenhuma (lembre-se "" é nenhuma) a linha 50 desvia o computador para a linha 40.

A linha 60 imprime a tecla que foi pressionada.

A linha 70 desvia o computador para as linhas que imprimem o parágrafo quando for pressionada a tecla "/".

A linha 80 cria um string e indexa com o número X, que é igual a 1 até que você pressione um ponto "." . Então, a linha 90 faz X igual a X+1.

Por exemplo, se a primeira letra pressionada for "B",



A$(1) SERÁ IGUAL A "B"

Se a segunda letra pressionada for "A",

A$(1) SERÁ IGUAL A A$(1) - QUE É "B" + "A" ou "BA"

E assim por diante, até que A$(1) seja igual a "BATATINHA QUANDO NASCE SE ESPARRAMA PELO CHAO". Neste instante, você pressiona "." e A$(1) será igual a toda a sentença - "BATATINHA QUANDO NASCE SE ESPARRAMA PELO CHAO." A próxima letra que for pressionada pertencerá a A$(2).

As linhas 140 a 160 imprimem todo o parágrafo.

UM PROCESSADOR DE PALAVRAS - DESAFIO

Aqui está um programa difícil (... mas não impossível! ...) para aqueles que ficaram intrigados com processadores de palavras. Transforme este programa em um processador de palavras muito versátil e que faça as seguintes tarefas:

1 - imprima as sentenças que você quiser
2 - permita alterar as sentenças

Você terá que rever o programa desafio que fizemos no fim do capítulo 12. O nosso programa encontra-se no fim deste manual.

FAÇA VOCÊ MESMO O SEU PROGRAMA

PARA AQUELES QUE TÊM IMPRESSORA

Se você tem uma impressora, provavelmente já está apto a usá-la. Conecte o cabo da impressora no terminal marcado "RS - 232C" na parte de trás do computador. Ligue-a e coloque o papel contínuo. Se você tiver alguma dúvida, consulte o manual da impressora, ele ensina como prepará-la.

Pronto? Digite este pequeno programa:

```
10 INPUT A$
20 PRINT#-2, A$
```

Agora digite:

LLIST (ENTER)

... e observe a impressora funcionar. Isto certamente vai facilitar a listagem de longos programas! Se o seu programa não for "listado" na impressora, verifique se ela está ligada e conectada corretamente. Tente LLIST novamente.

Agora execute o programa... forneça qualquer informação e veja a impressora trabalhar.

PRINT#-2 diz ao computador para imprimir, não na tela, mas sim no equipamento #-2, que é a impressora.

Pressione a tecla (SHIFT) e (0) (zero) simultaneamente de modo que as letras que forem digitadas apareçam na tela em cores invertidas (verdes com fundo preto). Você está agora escrevendo com letras maiúsculas e minúsculas. As cores invertidas (reversas) representam, na verdade, letras minúsculas (e não grifadas).

Digite uma letra enquanto estiver pressionando a tecla (SHIFT). Ela aparecerá em cores normais o que quer dizer que é uma letra maiúscula.

Todas as letras do comando RUN devem aparecer em cores normais.

Pressione a tecla (SHIFT) e digite o comando RUN ao mesmo tempo:

RUN (ENTER)

Escreva uma sentença com letras maiúsculas e minúsculas.

A MINHA IMPRESSORA IMPRIME LETRAS MINUSCULAS

Uma impressora com letras maiúsculas e minúsculas complementam um processador de palavras. Dê uma olhada no programa que fizemos no início deste capítulo. Que alterações faria você nas linhas 140 a 160 para que o parágrafo fosse impresso na impressora ao invés da tela?

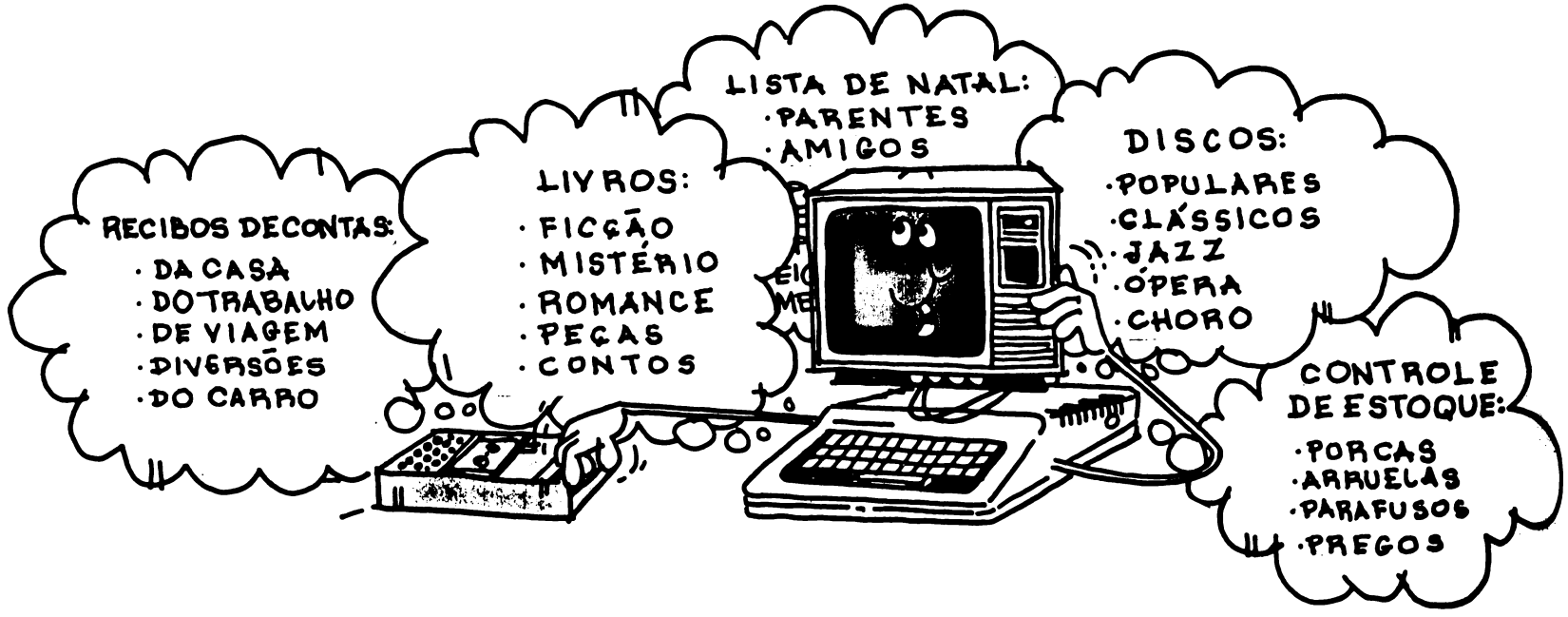
Já achou a resposta? Você simplesmente alteraria a linha 150 para:

```
150 PRINT #-2, A$(Y);
```

ASSUNTOS APRENDIDOS NO CAPÍTULO 21

PALAVRAS BASIC	CONCEITO BASIC
LLIST	variáveis strings indexadas
PRINT # -2	

CAPÍTULO 22



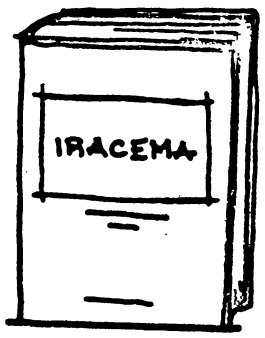
GRAVE A SUA COLEÇÃO DE LIVROS

(OU DISCOS, SELOS, RECIBOS DE CONTAS ...)

22

GRAVE A SUA COLEÇÃO DE LIVROS

(OU DISCOS, SELOS, RECIBOS DE CONTAS...)



Você já aprendeu a gravar programas numa fita cassete. Agora, mostraremos como usá-la para gravar e organizar listas de dados. Quando a lista estiver gravada na fita, você poderá usar a capacidade do computador de fazer inclusões, alterações, análises e impressões sempre que necessário.

Pronto para se organizar? Vamos começar com os seus livros. Aqui está uma pequena lista:

1 - O CORTIÇO
2 - IRACEMA
3 - A ROSA DO POVO
4 - EU ROBÔ

Para gravar esta lista numa fita e depois lê-la de volta para a memória do computador, você vai precisar de um programa. Será preciso digitar todo o programa antes de ver como

ele funciona, por isso, tenha um pouco de paciência.

A letra "O" indica que o arquivo está sendo aberto para saída de dados (OUTPUT)

Comece digitando:

```
10 OPEN "O", # -1, "LIVROS"
```

Isto diz ao computador para abrir um canal de comunicação com o periférico #-1 que é o gravador de fita cassete. Neste programa vamos armazenar dados num arquivo chamado LIVROS.

Um arquivo é um conjunto de informações — tais como títulos de livros — armazenadas com o mesmo nome.

Agora digite:

```
15 CLS: PRINT "FORNECA OS TITULOS DOS LIVROS— DIGITE<XX>QUANDO TERMINAR"
20 INPUT "TITULO"; T$
30 PRINT #-1, T$
40 GOTO 15
```

Isto permite que você forneça T$, que conterá os títulos dos livros um a um. Cada vez que você fornecer um título, o computador o imprimirá — não na tela, mas sim, no periférico #-1 que é o gravador cassete.

Acrescente estas linhas:

```
25 IF T$="XX" THEN 50
50 CLOSE #-1
```

Isto permite que você digite XX quando tiver terminado de fornecer todos os títulos dos livros. O computador então fechará o canal de comunicação com o periférico # -1 (o gravador de fita cassete).

Acrescente mais estas três linhas:

```
1 CLS
2 PRINT "POSICIONE A FITA E PRESSIONE AS TECLAS PLAY E RECORD"
4 INPUT "PRESSIONE < ENTER > QUANDO ESTIVER PRONTO"; R$
```

Na verdade, você pode gravar em qualquer posição da fita.

Aí está o programa, mas antes de executá-lo é necessário:

ATENÇÃO: Verifique se a fita está posicionada na parte magnética, e não na banda inicial de plástico.

- conectar o gravador de fita cassete. No capítulo 8 mostramos como fazê-lo
- colocar a fita no gravador e rebobiná-la
- pressionar as teclas PLAY e RECORD simultaneamente

Pronto? Verifique se o seu programa está igual ao nosso:

```
1  CLS
2  PRINT "POSICIONE O GRAVADOR E PRESSIONE AS TECLAS PLAY E RECORD"
4  INPUT "PRESSIONE < ENTER > QUANDO ESTIVER  PRONTO"; R$
10 OPEN "O", # -1, "LIVROS"  <---- abertura do arquivo
15 CLS: PRINT "FORNECA OS TITULOS DOS LIVROS - DIGITE<XX>QUANDO TERMINAR"
20 INPUT "TITULO"; T$
25 IF T$="XX" THEN 50
30 PRINT # -1, T$  <---- gravação dos títulos na fita
```

```
40 GOTO 15
50 CLOSE #-1 <------ fechamento do arquivo
```

Está tudo correto? ... Execute-o.

Note que assim que você pressionar < ENTER > , o motor do gravador entrará em funcionamento. O computador estará abrindo (OPEN) um arquivo em fita e chamando-o de "LIVROS".

Quando ele pedir, forneça os quatro títulos sugeridos acima e então digite < XX > :

O computador limpará a tela após cada título.

```
TITULO? O CORTICO
TITULO? IRACEMA
TITULO? A ROSA DO POVO
TITULO? EU ROBO
TITULO? XX
```

Sempre que você fornecer um título, o computador o colocará numa área especial da memória reservada para o gravador cassete. Quando você digitar XX indicando o fim da lista, o motor do gravador será acionado novamente e todos os dados serão transferidos para a fita (linha 30) e então, por último, na linha 50, o computador fechará o canal de comunicação com o gravador cassete.

Agora, todos os títulos estão na fita cassete e para carregá-los de volta à memória do computador digite:

```
60 CLS: PRINT "REBOBINE A FITA E PRESSIONE PLAY"
```

A letra "I" indica que o arquivo está sendo aberto para entrada de dados (INPUT).

```
70 INPUT "PRESSIONE < ENTER > QUANDO ESTIVER PRONTO"; R$
80 OPEN "I", #-1, "LIVROS"
```

A linha 80 abre um canal de comunicação para um arquivo chamado LIVROS e que se encontra numa fita cassete. Agora, ao invés de abrir o canal para saída (OUTPUT), ele está sendo aberto para entrada (INPUT) de dados.

Acrescente estas linhas:

```
90  INPUT #-1, T$
100 PRINT T$
```

A linha 90 lê o primeiro título (T$). Desta vez, a entrada não é feita pelo teclado, mas sim, a partir do gravador cassete. A linha 100 imprime T$ na tela.

Agora acrescente estas linhas:

```
85  IF EOF(-1) THEN 120
110 GOTO 85
120 CLOSE #-1
```

A linha 85 diz que se você chegou ao fim do arquivo LIVROS, então vá para a linha 120,que fecha o canal de comunicação com o gravador.

ERRO: Verifique se você colocou a linha EOF(-1) antes da linha INPUT #-1; caso contrário, ocorrerá um erro IE - tentativa de ler dados depois do fim do arquivo.

Veja esta última parte do programa digitando:

```
LIST 60-
```

Você deve ter a seguinte listagem na tela:

```
60 CLS: PRINT "REBOBINE A FITA E PRESSIONE PLAY"
70 INPUT "PRESSIONE<ENTER>QUANDO ESTIVER PRONTO"; R$
80 OPEN "I", #-1, "LIVROS"
85 IF EOF(-1) THEN 120
90 INPUT #-1, T$
100 PRINT T$
110 GOTO 85
120 CLOSE #-1
```

abertura do arquivo

leitura dos títulos da fita

fechamento do arquivo

Agora execute-o. Digite:

```
RUN 60
```

Certifique-se de pressionar apenas a tecla PLAY, e não RECORD. Verifique também se a fita foi rebobinada.

Siga as instruções do computador...

Quando você pressionar (ENTER), repare que o motor do gravador é acionado. O computador está lendo os dados da fita cassete e logo que terminar a leitura, ele imprimirá os quatro itens na tela.

Que tal uma rápida revisão do programa?

Se o computador ficar "preso" quando estiver se comunicando com o gravador, basta pressionar o botão RESET para recuperar o controle sobre ele. Em seguida, verifique se não estão faltando linhas ou se alguma delas não foi digitada incorretamente.

```
1   CLS
2   PRINT "POSICIONE O GRAVADOR E PRESSIONE AS TECLAS PLAY E RECORD"
4   INPUT "PRESSIONE < ENTER > QUANDO ESTIVER PRONTO"; R$

10  OPEN "O",#-1,"LIVROS"
15  CLS: PRINT "FORNECA OS TITULOS DOS LIVROS - DIGITE < XX > QUANDO TERMINAR"
20  INPUT "TITULO"; T$
25  IF T$="XX" THEN 50
30  PRINT #-1, T$
40  GOTO 15
50  CLOSE #-1

60  CLS: PRINT "REBOBINE A FITA E PRESSIONE PLAY"
70  INPUT "PRESSIONE < ENTER > QUANDO ESTIVER PRONTO"; R$

80  OPEN "I", #-1, "LIVROS"
85  IF EOF(-1) THEN 120
90  INPUT #-1, T$
100 PRINT T$
110 GOTO 85
120 CLOSE #-1
```

rotina que grava os títulos na fita

rotina que lê os títulos da fita

Na linha 30 gravamos T$ (o título do livro) na fita. Para fazê-lo foi preciso abrir um canal de comunicação com o gravador para saída de dados. Após o término da gravação, fechamos este canal de comunicação.

Na linha 90 lemos T$ a partir da fita. Para tanto, abrimos um canal de comunicação com o gravador para entrada de dados. Após o término da leitura, fechamos este canal de comunicação.

Você entendeu? Pense um pouco e responda a estas três perguntas...

P E R G U N T A S

1. O que ocorreria se você não incluísse a linha 50 e executasse o programa?

RESPOSTA: Sem a linha 50, o canal de comunicação com o gravador permaneceria aberto para saída de dados (OUTPUT), e o computador não poderia abrí-lo novamente para entrada de dados (INPUT). Neste caso, a linha 80 causaria um erro AO - tentativa de abrir um arquivo que já se encontra aberto.

2. Daria certo se eliminássemos as linhas 50 e 80 e executássemos o programa?

RESPOSTA: Isto também não funcionaria, porque sem as linhas 50 e 80, o canal de comunicação permanece aberto para saída de dados (OUTPUT) e quando a linha 90 pedir ao computador que leia um dado do gravador, ocorrerá um erro NO - arquivo não aberto. O arquivo não está aberto para entrada de dados (INPUT).

3. Daria certo se modificássemos as linhas 90 e 100 para:

```
90  INPUT #-1, X$
100 PRINT X$
```

RESPOSTA: Sim, vai dar certo porque o computador não leva em consideração o fato de você ter chamado os títulos de T$ quando os gravou na fita. Quando eles são lidos para a memória, o computador simplesmente procura um string na fita e o chama de X$.

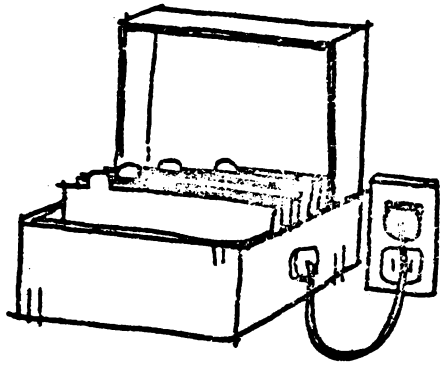
UM ARQUIVO ELETRÔNICO

Como vão suas ambições? Que tal alterar o programa de forma que seja possível gravar todos estes dados na fita cassete:

TÍTULO	AUTOR	ASSUNTO
O CORTIÇO	ALUÍSIO AZEVEDO	ROMANCE
IRACEMA	JOSÉ DE ALENCAR	ROMANCE
A ROSA DO POVO	CARLOS D. DE ANDRADE	POESIA
EU ROBÔ	ISAAC ASIMOV	FICÇÃO CIENTÍFICA

Vamos inicialmente trabalhar na primeira metade do programa anterior — a parte que trata de saída de dados para a fita cassete. Acrescente estas linhas ao programa:

```
26 INPUT "AUTOR"; A$
28 INPUT "ASSUNTO"; S$
29 IF A$="XX" OR S$="XX" THEN 50
```



Para gravar todos estes dados na fita cassete, simplesmente altere a linha 30:

```
30 PRINT #-1, T$, A$, S$
```

Agora, vamos mexer na segunda metade do programa. Como modificaria você as linhas 90 e 100 para que o computador lesse dados da fita e imprimisse o título, o autor e o assunto?

EXERCÍCIO DE PROGRAMAÇÃO

Aqui está nossa resposta:

```
90  INPUT #-1, T$, A$, S$
100 PRINT "TITULO :"    T$
102 PRINT "AUTOR  :"    A$
104 PRINT "ASSUNTO:"    S$
```

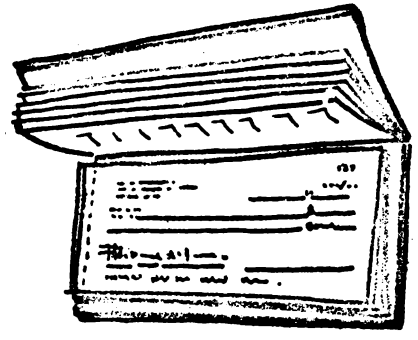
Como já foi dito anteriormente, não é necessário usar os mesmos nomes de variáveis usadas na gravação. A forma abaixo também daria certo:

```
90  INPUT # -1, X$, Y$, Z$
100 PRINT "TITULO  :"   X$
102 PRINT "AUTOR   :"   Y$
104 PRINT "ASSUNTO :"   Z$
```

ESCOLHA UM ASSUNTO

Agora, você pode tirar vantagem de toda esta organização. Por exemplo, você pode pedir que o computador imprima uma lista de livros de um assunto pré-determinado.

Acrescente estas linhas:



```
130 CLS
140 INPUT "INDIQUE O ASSUNTO"; C$
150 PRINT "REBOBINE A FITA E PRESSIONE PLAY"
160 INPUT "PRESSIONE < ENTER > QUANDO ESTIVER PRONTO"; E$
170 CLS: PRINT "LIVROS DE " C$ : PRINT
180 OPEN "I", # -1, "LIVROS"
190 IF EOF(-1) THEN 230
200 INPUT # -1, T$, A$, S$
210 IF S$=C$ THEN PRINT T$, A$
```

```
220 GOTO 190
230 CLOSE #-1
```

digite RUN 130 para executar o programa. Se você escolher ROMANCE, a execução do programa se daria da seguinte forma:

```
INDIQUE O ASSUNTO? ROMANCE
REBOBINE A FITA E PRESSIONE PLAY
PRESSIONE < ENTER > QUANDO ESTIVER PRONTO

LIVROS DE ROMANCE

O CORTICO    ALUISIO AZEVEDO
IRACEMA      JOSE' DE ALENCAR
```

CONTROLE DE TALÃO DE CHEQUES

Agora é a sua vez de tentar. Suponha que você tenha estes cheques:

Nº	DATA	PAGO A	CONTA	VALOR(CR$)
101	20/5	SUPERMERCADO	COMIDA	5.000,00
102	20/5	OFICINA	CARRO	12.000,00
103	22/5	RESTAURANTE	COMIDA	3.000,00
104	25/5	HOTEL	FÉRIAS	30.000,00
105	26/5	FARMÁCIA	DOENÇA	3.000,00

Escreva um programa que grave todos estes dados numa fita cassete e em seguida, faça uma leitura de forma que quando for fornecida uma conta de despesa — comida por exemplo — o computador indique o total da despesa desta conta.

Lembre-se como gravar informações numa fita cassete:

REGRAS PARA GRAVAR DADOS NUMA FITA

Para gravar dados numa fita, é preciso:

1 - Abrir um canal de comunicação com o gravador para saída de dados.

2 - Gravar o dado na fita.

3 - Continuar a gravação de dados até o fim destes, e em seguida

4 - Fechar o canal de comunicação com o gravador

e como ler informações da fita cassete:

REGRAS PARA LER DADOS DE UMA FITA

Para ler dados de uma fita,é preciso:

1 - Abrir um canal de comunicação com o gravador para entrada de dados.

2 - Usar EOF(-1) para verificar se foi atingido o fim do arquivo na fita

3 - Ler o dado da fita

4 - Continuar lendo dados da fita até que você encerre a leitura ou atinja o fim do arquivo

5 - Fechar o canal de comunicação com o gravador.

FAÇA VOCÊ MESMO O SEU PROGRAMA

Aqui está a nossa resposta:

```
5   CLS:PRINT "POSICIONE O GRAVADOR E PRESSIONE AS TECLAS PLAY E RECORD"
7   INPUT "PRESSIONE < ENTER > QUANDO ESTIVER PRONTO"; R$
10  OPEN "O", #-1, "CHEQUES"
15  CLS: PRINT "FORNECA OS CHEQUES E DIGITE < XX > QUANDO TERMINAR"
20  INPUT "NUMERO:"; N$
25  IF N$="XX" THEN 90
30  INPUT "DATA :"; D$
40  INPUT "PAGO  A:"; P$
50  INPUT "CONTA:"; C$
60  INPUT "VALOR:CR$"; V
70  PRINT #-1, N$, D$, P$, C$, V
80  GOTO 15
90  CLOSE #-1
92  CLS: T=0
```

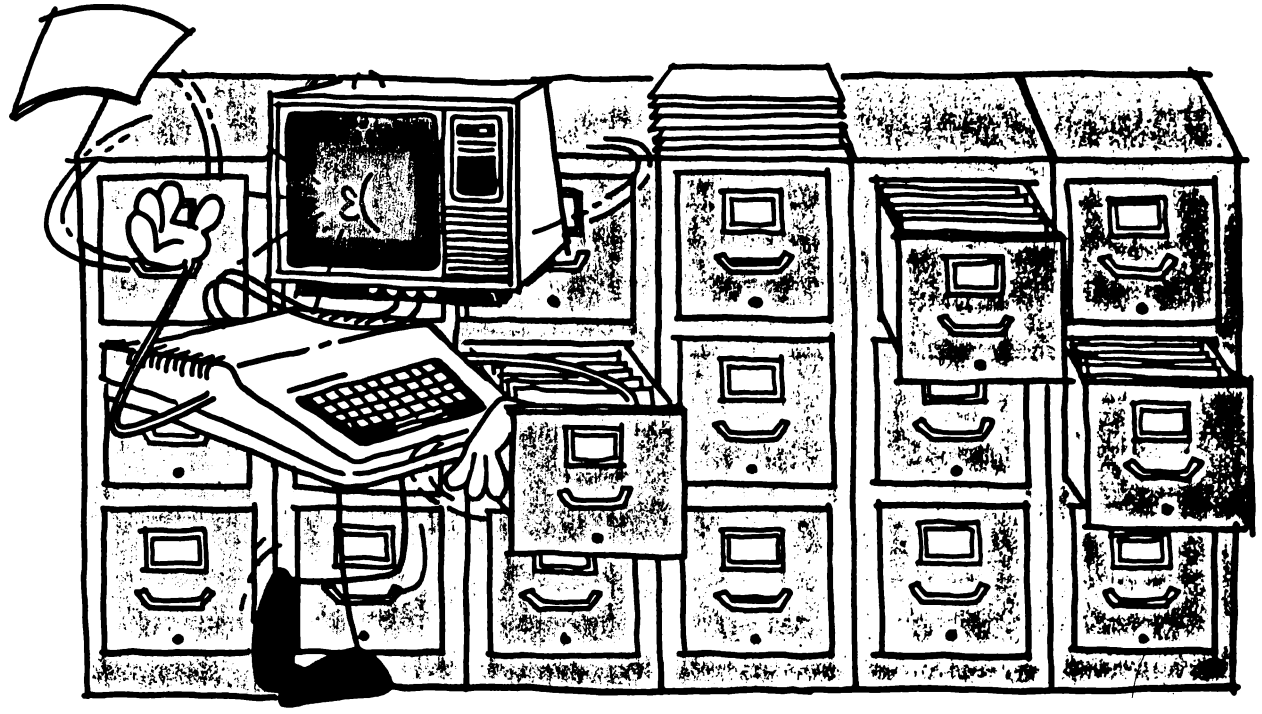
```
95  INPUT "INDIQUE A CONTA"; B$
100 PRINT "REBOBINE A FITA E PRESSIONE PLAY"
110 INPUT "PRESSIONE < ENTER > QUANDO ESTIVER PRONTO"; R$
120 OPEN "I", #-1, "CHEQUES"
130 IF EOF(-1) THEN 170
140 INPUT #-1, N$, D$, P$, C$, V
150 IF B$=S$ THEN T=T+V
160 GOTO 130
170 CLOSE #-1
180 PRINT "A DESPESA TOTAL DA CONTA ";B$;" FOI CR$" T
```

Agora que você entendeu como se guarda informações numa fita cassete, seria interessante dar uma olhada em alguns dos programas simples no fim deste manual. Eles lhes darão algumas outras idéias de como usar arquivos em fita.

ASSUNTOS APRENDIDOS NO CAPÍTULO 22

PALAVRAS BASIC	CONCEITO BASIC
OPEN	ARQUIVOS DE DADOS
CLOSE	
PRINT #-1	
INPUT #-1	
EOF	

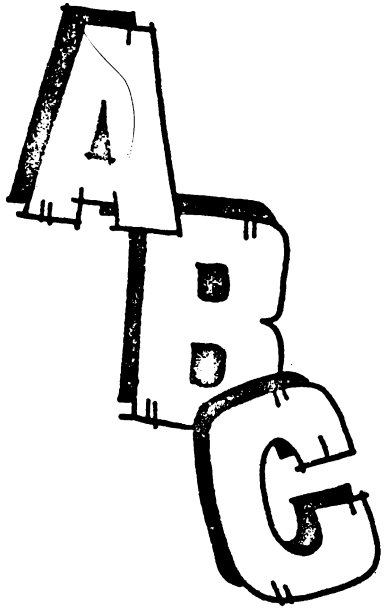
CAPÍTULO 23



ARQUIVAR — TÃO SIMPLES QUANTO O ABC

23

ARQUIVAR – TÃO SIMPLES QUANTO O ABC



Qualquer arquivista lhe dirá que é muito mais simples encontrar coisas se elas estiverem em ordem alfabética. Neste caso, porque não deixar o computador ter este trabalho de ordenar? Digite este programa:

```
10 INPUT "DIGITE DUAS PALAVRAS"; A$, B$
20 IF A$ < B$ THEN PRINT A$ "VEM ANTES DE" B$
30 IF A$ > B$ THEN PRINT B$ "VEM ANTES DE" A$
40 IF A$ = B$ THEN PRINT "AS DUAS PALAVRAS SAO IGUAIS"
50 GOTO 10
```

e execute-o. Continue fornecendo palavras até que se convença que o computador sabe colocar em ordem alfabética.

Quando se trata de strings, os sinais de "maior que", "menor que" e "igual" discutidos no capítulo 11 passam a ter outros significados. Eles dizem qual dos dois strings vem primeiro em ordem alfabética.

< precede alfabeticamente
<= precede ou é idêntico alfabeticamente
> sucede alfabeticamente
>= sucede ou é idêntico alfabeticamente
= é idêntico

Já que o computador sabe colocar em ordem alfabética, você pode escrever um programa para ordenar alfabeticamente uma longa lista de palavras. Aqui está o nosso programa:

Você pode fazer com que o computador ordene mais palavras, alterando nas linhas 10, 20, 70 e 90 o número 5 para outro valor qualquer.

```
10  DIM A$(5)
20  FOR I=1 TO 5
30  INPUT "FORNECA UMA PALAVRA"; A$(I)
40  NEXT I
50  X=0
60  X=X+1
70  IF X>5 THEN GOTO 70
80  IF A$(X)="ZZ" THEN 60
90  FOR Y=1 TO 5
100 IF A$(Y) < A$(X) THEN X=Y
110 NEXT Y
120 PRINT A$(X)
130 A$(X)="ZZ"
140 GOTO 50
```

Digite e execute este programa.

Antes de explicarmos como ele funciona, mostraremos o que ocorre quando ele é executado. Digite:

```
30  READ A$(I)
200 DATA MIGUEL, TIAGO, DAVI, ALEX, SUSANA
```

De forma que ordenaremos as mesmas palavras. Elimine a linha 120 e digite:

```
120
5   CLS
35  PRINT A$(I)
85  V=V+1
105 PRINT @  15+32 * (V-1), A$(X)
135 GOSUB 500
500 FOR I=1 TO 5
510 PRINT @  0+32 * (I-1), A$(I);
520 NEXT I
530 RETURN
```

Estas linhas permitem que você acompanhe o que está sendo feito pelo programa. Não é preciso estudá-las, apenas digite-as como estão. Elas não interferem no processo de ordenação.

Execute o programa.

Está muito rápido? Digite esta linha, ela vai tornar o programa mais lento e você poderá ver o que está ocorrendo:

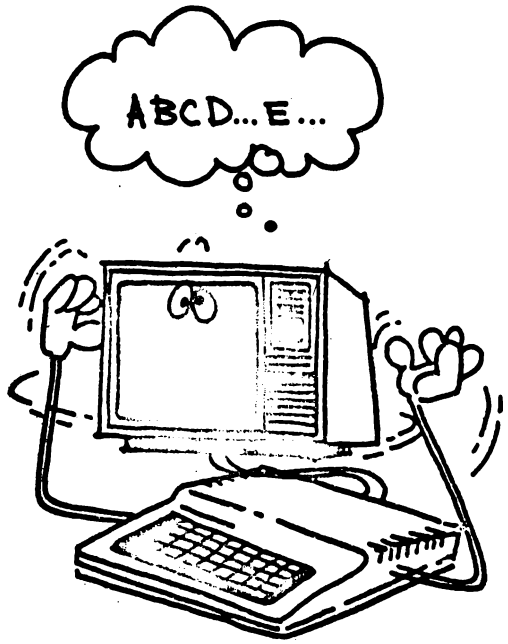
```
107 FOR T=1 TO 600: NEXT T
```

PRIMEIRA POSIÇÃO

MIGUEL TIAGO DAVI ALEX SUSANA	MIGUEL	
MIGUEL TIAGO DAVI ALEX SUSANA	MIGUEL	
MIGUEL TIAGO DAVI ALEX SUSANA	MIGUEL	
MIGUEL TIAGO DAVI ALEX SUSANA	DAVI	
MIGUEL TIAGO DAVI ALEX SUSANA	ALEX	
MIGUEL TIAGO DAVI ZZ SUSANA	ALEX	

SEGUNDA POSIÇÃO

MIGUEL TIAGO DAVI ZZ SUSANA	ALEX MIGUEL	
MIGUEL TIAGO DAVI ZZ SUSANA	ALEX MIGUEL	
MIGUEL TIAGO DAVI ZZ SUSANA	ALEX MIGUEL	
MIGUEL TIAGO DAVI ZZ SUSANA	ALEX DAVI	
MIGUEL TIAGO DAVI ZZ SUSANA	ALEX DAVI	
MIGUEL TIAGO ZZ ZZ SUSANA	ALEX DAVI	



Quando o programa começa, MIGUEL é comparado com MIGUEL para ver quem precede o outro alfabeticamente. MIGUEL permanece no topo. MIGUEL é então comparado com TIAGO e novamente MIGUEL permanece no topo.

Em seguida MIGUEL é comparado com DAVI. Como DAVI precede MIGUEL, DAVI assume o lugar de MIGUEL no topo.

Agora DAVI é comparado com ALEX. Como ALEX precede DAVI, ele vai para o topo. Finalmente, ALEX é comparado com SUSANA e permanece no topo.

Agora que todos os nomes já foram comparados com a posição do topo, o computador repete o ciclo para determinar a segunda, a terceira, a quarta e a quinta posições. ALEX transforma-se em ZZ para não assumir outras posições.

Agora que você viu o que o programa faz, vamos percorrê-lo usando os mesmos nomes acima.

As linhas 50 e 60 ajustam o valor de X, que é igual a 1 na primeira passagem.

Em seguida as linhas 60 a 110 comparam A$(1) — MIGUEL — com todos os outros nomes no vetor A$, até encontrar uma palavra que preceda A$(1). No nosso exemplo, a terceira palavra — DAVI — é a encontrada. A linha 100 então faz com que A$(X) seja igualada a A$(3) — posição de DAVI no vetor. Quando DAVI é comparado com a quarta palavra — ALEX — , A$(X) é igualada a A$(4).

A linha 120 imprime A$(4) — ALEX — e a linha 130 iguala A$(4) a ZZ. Neste ponto, as linhas 50 e 60 fazem X=1 novamente. A$(1) — MIGUEL — é novamente comparado com os outros no

mes no vetor.

Quando a posição de MIGUEL no vetor se transforma em ZZ, a linha 80 desvia o computador de volta para a linha 60, que faz com que X seja igual a 2. A$(2) — TIAGO — é então comparado com todos os nomes no vetor.

Quando todas as posições do vetor contiverem ZZ, a linha 70 termina o programa.

Usando esta rotina de ordenação (SORT), modifique o programa do capítulo anterior, de forma que o computador ordene todos os livros pelo título, autor ou assunto.

FAÇA VOCÊ MESMO O SEU PROGRAMA

Nossa resposta está no fim deste manual.

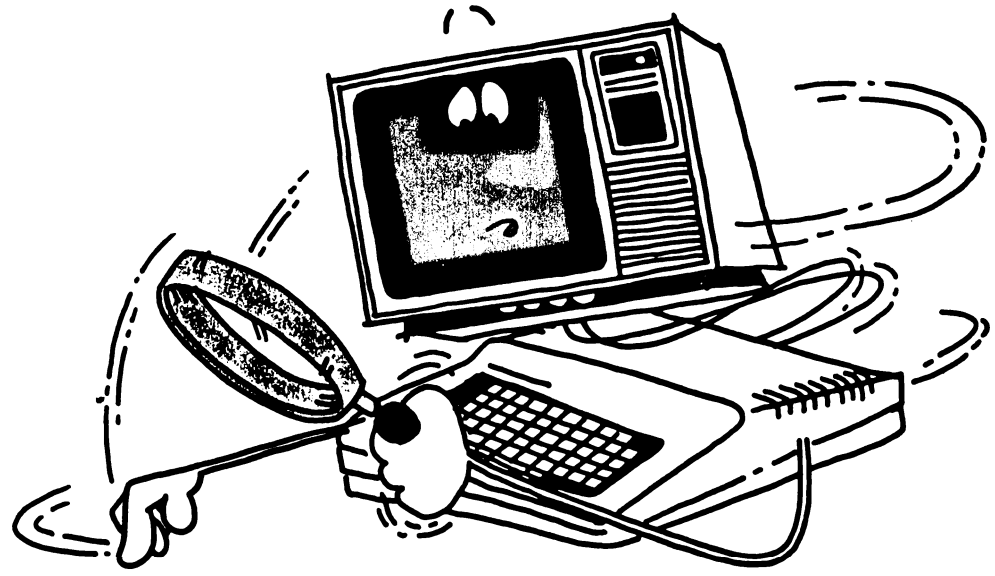
O método de ordenação (SORT) que mostramos é um dos mais fáceis de se programar. Existem outros métodos mais complexos que são muito mais rápidos. Se você tiver um número muito grande de itens para ordenar, talvez tenha interesse em estudar outros métodos.

ASSUNTOS APRENDIDOS NO CAPÍTULO 23

SÍMBOLOS BASIC

>
<
=

CAPÍTULO 24



SENDO ANALÍTICO

24

SENDO ANALÍTICO

Usaremos apenas três municípios para simplificar.

Desta vez vamos chamá-los de candidatos 1 e 2, ao invés de A e B.

Existe uma maneira bem fácil de analisar qualquer coisa. Se você indexar todas as suas informações, é possível vê-las de diferentes pontos de vista.

Por exemplo, vamos utilizar o programa de apuração de votos do capítulo 20. Aqui estão os dados:

R E S U L T A D O S D A E L E I Ç Ã O

MUNICÍPIO	VOTOS APURADOS PARA O CANDIDATO 1	VOTOS APURADOS PARA O CANDIDATO 2
1	143	678
2	215	514
3	125	430

Se rotularmos cada item com 2 índices, poderemos colocá-los em uma variável indexada (matriz), da seguinte forma:

No capítulo 20 criamos uma variável indexada A para o candidato 1 e uma variável indexada B para o candidato B. Agora estamos juntando tudo em uma única variável indexada, com dois índices, chamada V.

M A T R I Z V

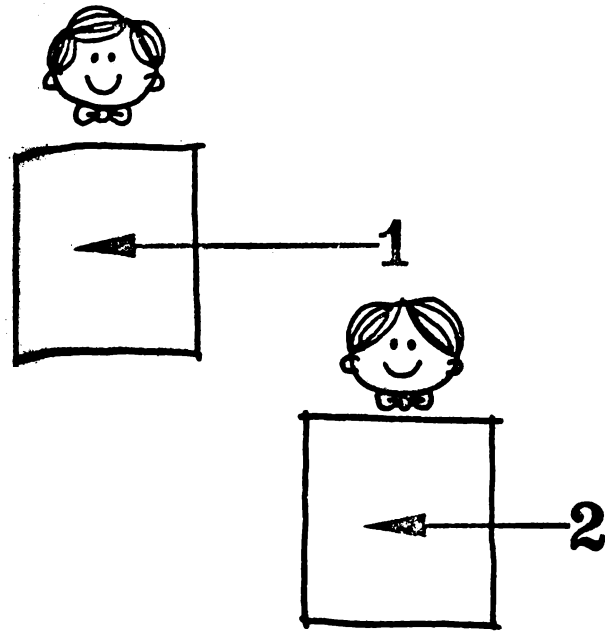
	CANDIDATO 1	CANDIDATO 2
MUNICÍPIO 1	V(1,1)	V(1,2)
	143	678
MUNICÍPIO 2	V(2,1)	V(2,2)
	215	514
MUNICÍPIO 3	V(3,1)	V(3,2)
	125	430

O primeiro índice representa o município de onde vêm os votos e o segundo indica para que candidato são os votos.

Por exemplo, usamos V(1,2) para rotular os 678 votos, e com isso determinamos que os 678 votos estão armazenados em uma matriz de nome V. Estes votos são do município 1 e do candidato 2.

Vamos colocar tudo isto num programa. Digite:

```
5   DIM V(3,2)
10  DATA 143, 678, 215, 514, 125, 430
20  FOR M=1 TO 3
30  FOR C=1 TO 2
40  READ V(M,C)
```

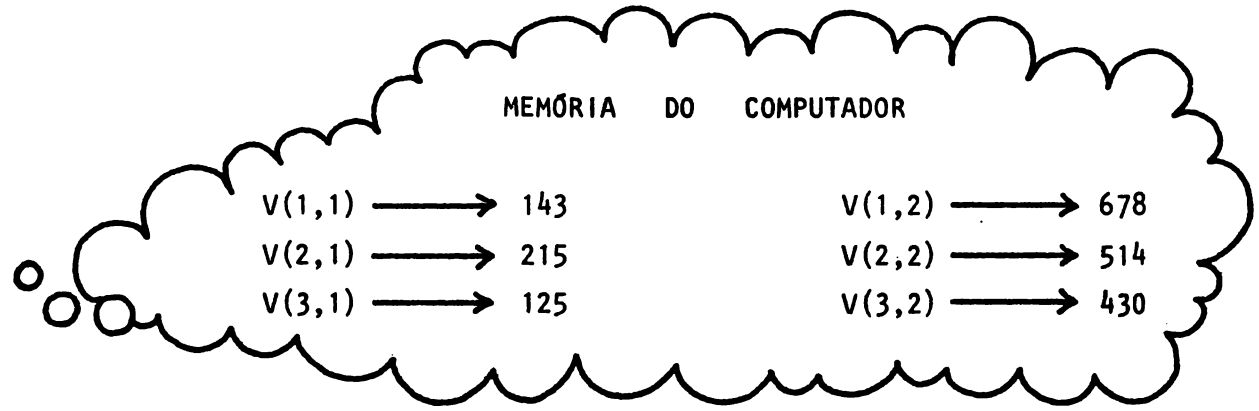



```
50  NEXT C
60  NEXT M
70  INPUT "NUMERO DO MUNICIPIO (1-3)"; M
80  IF M<1 OR M > 3 THEN 70
90  INPUT "NUMERO DO CANDIDATO (1-2)"; C
100 IF C < 1 OR C > 2 THEN 90
110 PRINT V(M,C)
120 GOTO 70
```

A linha 5 reserva espaço na memória para uma variável de nome V. Cada item tem 2 índices. O primeiro índice não pode ser maior que 3 e o segundo não pode ser maior que 2.

Execute o programa. Certifique-se de que todos os votos foram rotulados como anteriormente. Experimente várias combinações para os números dos municípios e dos candidatos.

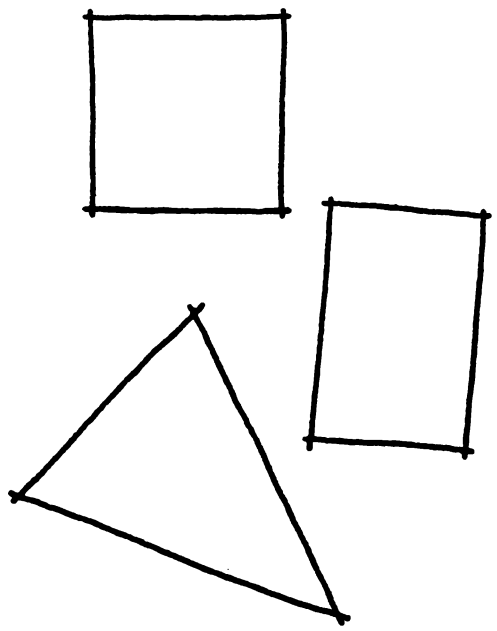
As linhas 20 a 60 lêem os votos para os candidatos 1 e 2 nos municípios 1, 2 e 3 e os armazenam na memória do computador.



Você ainda se lembra como eliminar linhas?

70 (ENTER)

elimina a linha 70.



Agora que você tem duas perspectivas diferentes para os grupos de votos, é só tirar vantagem disto. Elimine as linhas 70 a 120 e digite:

```
70   INPUT "DIGITE < 1 > PARA MUNICIPIO OU < 2 > PARA CANDIDATO"; R
80   IF R<1 OR R>2 THEN 70
100  ON R GOSUB 1000, 2000
110  GOTO 70

1000 INPUT "NUMERO DO MUNICIPIO (1-3)"; M
1010 IF M<1 OR M>3 THEN 1000
1015 CLS
1020 PRINT @ 132, "VOTOS NO MUNICIPIO" M
1030 PRINT
1040 FOR C=1 TO 2
1050 PRINT "CANDIDATO" C,
1060 PRINT V(M,C)
1070 NEXT C
1080 RETURN

2000 INPUT "NUMERO DO CANDIDATO (1-2)"; C
2010 IF C<1 OR C>2 THEN 2000
2015 CLS
2020 PRINT @ 132, "VOTOS PARA O CANDIDATO" C
2030 PRINT
2040 FOR M=1 TO 3
2050 PRINT "MUNICIPIO" D,
```

```
2060 PRINT V(M,C)
2070 NEXT M
2080 RETURN
```

Execute o programa e analise as informações a seu gosto.

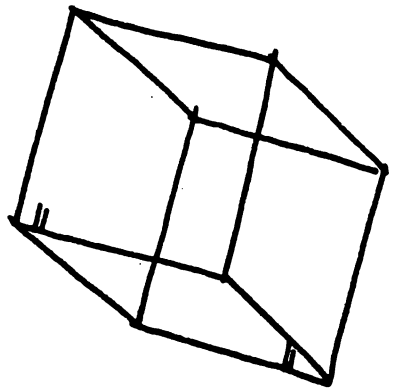
O que você acaba de programar é uma variável indexada bidimensional, ou seja, uma matriz. Ela é chamada assim porque todos os seus itens têm dois índices — número do município e do candidato. Nos capítulos anteriores, utilizamos apenas variáveis de uma dimensão (vetores) onde os itens tinham apenas um índice.

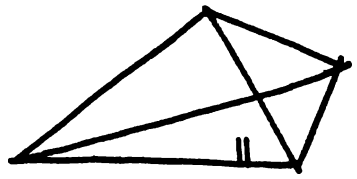
Se você for do tipo analítico, vai adorar o resto deste capítulo, caso contrário, não se aborreça, pule-o.

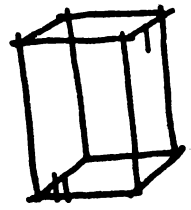
Não há nada que limite em dois o número de dimensões; este computador permite que você utilize variáveis indexadas com qualquer número de dimensões.

A TERCEIRA DIMENSÃO

Agora já estamos prontos para acrescentar grupos de interesse — uma terceira dimensão à variável indexada. Os resultados anteriores se referiam todos ao grupo de interesse 1. Também apuramos os resultados relativos aos grupos de interesse 2 e 3, mostrados a seguir.







VOTOS RELATIVOS AO GRUPO DE INTERESSE 1

	CANDIDATO 1	CANDIDATO 2
MUNICÍPIO 1	143	678
MUNICÍPIO 2	215	514
MUNICÍPIO 3	125	430

VOTOS RELATIVOS AO GRUPO DE INTERESSE 2

	CANDIDATO 1	CANDIDATO 2
MUNICÍPIO 1	525	54
MUNICÍPIO 2	318	157
MUNICÍPIO 3	254	200

VOTOS RELATIVOS AO GRUPO DE INTERESSE 3

	CANDIDATO 1	CANDIDATO 2
MUNICÍPIO 1	400	119
MUNICÍPIO 2	124	300
MUNICÍPIO 3	75	419

Aqui está como rotulamos todos estes votos na variável indexada V:

V A R I Á V E L I N D E X A D A V

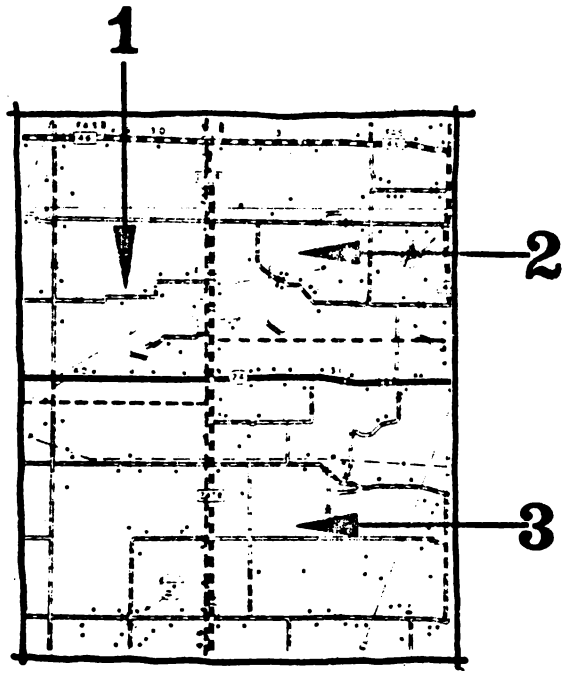
GRUPO DE INTERESSE 1

	CANDIDATO 1	CANDIDATO 2
MUNICÍPIO 1	V(1,1,1) 143	V(1,1,2) 678
MUNICÍPIO 2	V(1,2,1) 215	V(1,2,2) 514
MUNICÍPIO 3	V(1,3,1) 125	V(1,3,2) 430

GRUPO DE INTERESSE 2

	CANDIDATO 1	CANDIDATO 2
MUNICÍPIO 1	V(2,1,1) 525	V(2,1,2) 54
MUNICÍPIO 2	V(2,2,1) 318	V(2,2,2) 157
MUNICÍPIO 3	V(2,3,1) 254	V(2,3,2) 200

	GRUPO DE INTERESSE 3	
	CANDIDATO 1	CANDIDATO 2
MUNICÍPIO 1	V(3,1,1) 400	V(3,1,2) 119
MUNICÍPIO 2	V(3,2,1) 124	V(3,2,2) 300
MUNICÍPIO 3	V(3,3,1) 75	V(3,3,2) 419

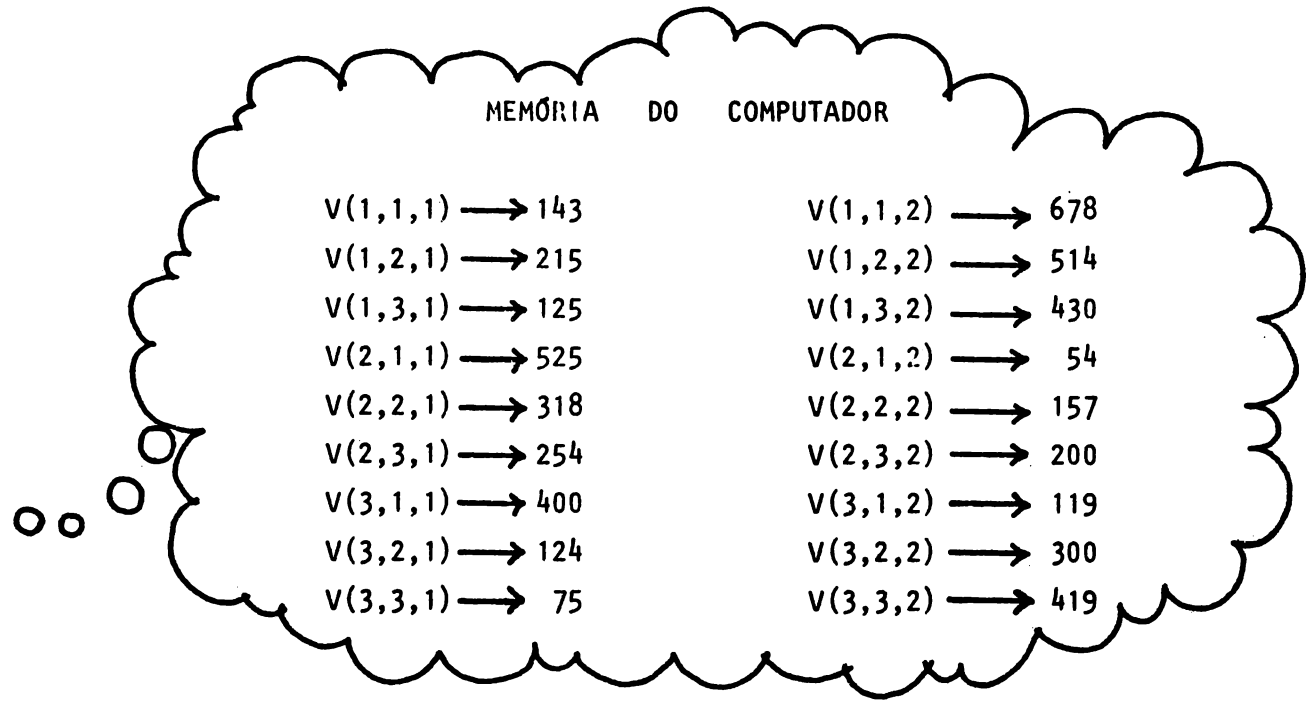


Para armazenar todos estes dados na memória do computador, limpe a memória (NEW) e digite:

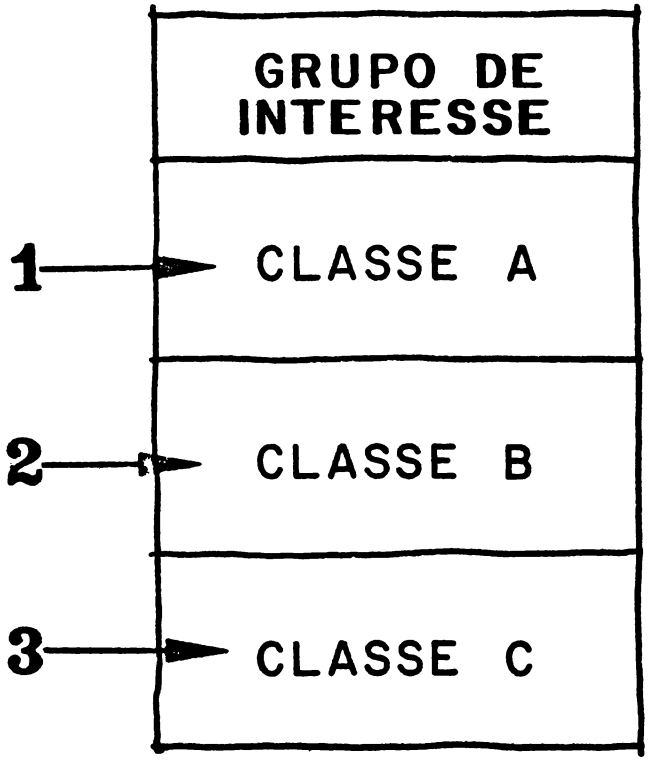
```
5   DIM V(3,3,2)
10  DATA 143, 678, 215, 514, 125, 430
20  DATA 525,  54, 318, 157, 254, 200
30  DATA 400, 119, 124, 300,  75, 419

40  FOR G=1 TO 3
50  FOR M=1 TO 3
60  FOR C=1 TO 2
70  READ V(G,M,C)
80  NEXT C
90  NEXT M
100 NEXT G
```

```
110 INPUT "NUMERO DO GRUPO DE INTERESSE (1-3)"; G
120 IF G<1 OR G>3 THEN 110
130 INPUT "NUMERO DO MUNICIPIO (1-3)"; M
140 IF M<1 OR M>3 THEN 130
150 INPUT "NUMERO DO CANDIDATO (1-2)"; C
160 IF C<1 OR C>2 THEN 150
170 PRINT V(G,M,C)
180 GOTO 110
```

Execute o programa e teste todos os índices. As linhas 40 até 100 colocam os seguintes dados na memória do computador:



Para melhor aproveitar as três dimensões, elimine as linhas 110 a 180 e digite:




```
110  PRINT: PRINT "DIGITE < 1 > PARA GRUPO"
120  PRINT "< 2 >PARA MUNICIPIO OU < 3 > PARA CANDIDATO"
130  P=224 : INPUT R
140  ON R GOSUB 1000, 2000, 3000
150  GOTO 110

1000 INPUT "GRUPO (1-3)"; G
1010 IF G <1 OR G >3 THEN 1000
1020 CLS
1030 PRINT @ 102, "VOTOS RELATIVOS AO GRUPO" G
1040 PRINT @ 168, "CAND. 1"
1050 PRINT @ 176, "CAND. 2"
1060 FOR M=1 TO 3
1070 PRINT @ P, "MUN." M
1080 FOR C=1 TO 2
1100 PRINT @ P+8*C, V(G,M,C),
1110 NEXT C
1120 P=P+32
1130 NEXT M
1140 RETURN

2000 INPUT "MUNICIPIO (1-3)"; M
2010 IF M<1  OR M  3 THEN 2000
2020 CLS
```

```
2030 PRINT @ 102, "VOTOS NO MUN." M
2040 PRINT @ 168, "CAND. 1"
2050 PRINT @ 176, "CAND. 2"
2060 FOR G=1 TO 3
2070 PRINT @ P, "GRUPO" G
2080 FOR C=1 TO 2
2100 PRINT @ P+8*C, V(G,M,C);
2110 NEXT C
2120 P=P+32
2130 NEXT G
2140 RETURN

3000 INPUT "CANDIDATO (1-2)"; C
3010 IF C <1 OR C >2 THEN 3000
3020 CLS
3030 PRINT @ 102, "VOTOS PARA O CAND." C
3040 PRINT @ 168, "MUN. 1"
3050 PRINT @ 176, "MUN. 2"
3060 PRINT @ 184, "MUN. 3"
3070 FOR G=1 TO 3
3080 PRINT @ P, "GRUPO" G
3090 FOR M=1 TO 3
3100 PRINT @ P+8*D, V(G,M,C);
3110 NEXT M
3120 P=P+32
3130 NEXT G
3140 RETURN
```

Escreva um programa que "dê cartas" utilizando uma variável indexada bidimencional (ma - triz). Uma das dimensões representará os quatro naipes e a outra o valor da carta (1-13).

FAÇA VOCÊ MESMO O SEU PROGRAMA

A nossa resposta está no fim deste manual.

ASSUNTOS APRENDIDOS NO CAPÍTULO 24

CONCEITO BASIC

variáveis indexadas
multi-dimensionais

APÊNDICES

APÊNDICE A

TONS MUSICAIS

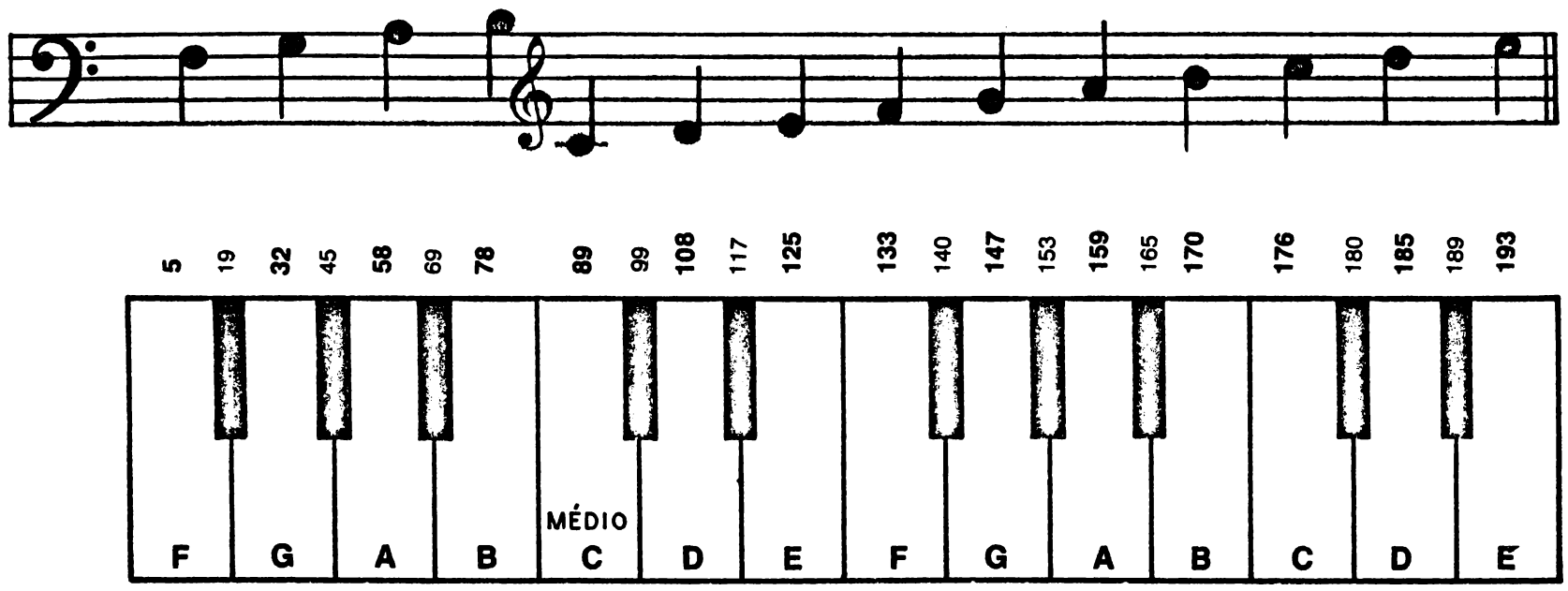
O computador pode tocar os tons musicais relacionados na próxima página. Você pode usar um teclado de piano ou outro instrumento musical qualquer para produzir músicas eletrônicas.

Se estiver usando um piano, o tom do computador para cada tecla, está indicado diretamente sobre a mesma. Por exemplo, o número do tom para o Dó médio (C) é 89.

Se estiver usando outro instrumento, o número do tom de cada nota está indicado embaixo dela. Por exemplo, o número do tom para:

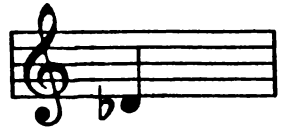
é 108.

APÊNDICE A



Se a nota é um semi-tom abaixo, selecione o número do tom imediatamente anterior.

Por exemplo:

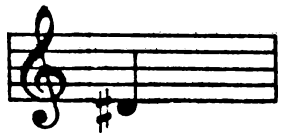


é 99.

APÊNDICE A

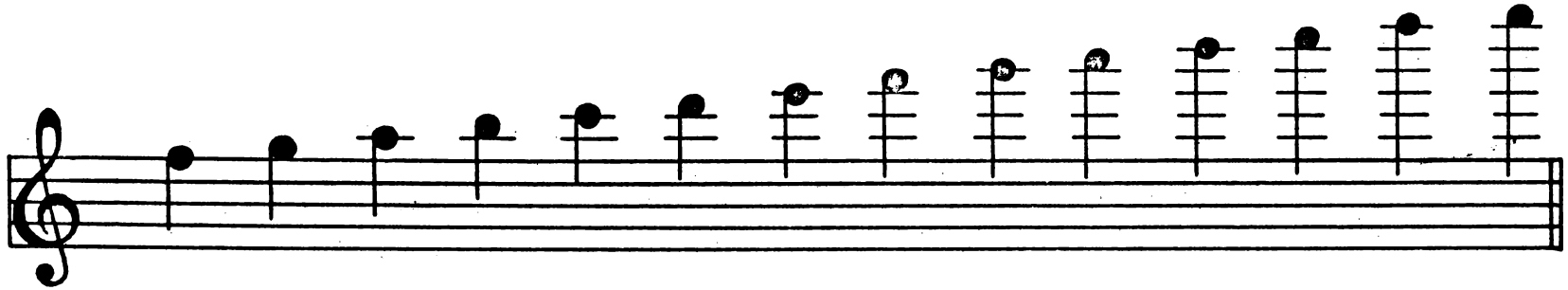
Se a nota é um semi-tom acima, selecione o número do tom imediatamente posterior.

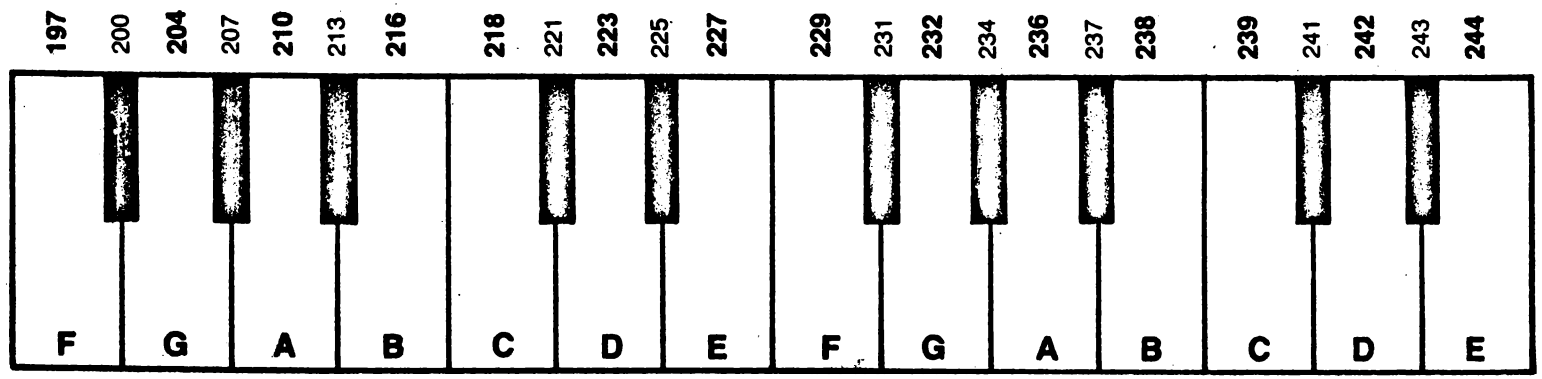
Por exemplo:



é 117.

No capítulo 5 mostramos como programar o computador para tocar uma canção.





APÊNDICE B

CORES E CARACTERES GRÁFICOS

C O R E S

A seguir apresentamos os códigos das cores que você pode mostrar na tela. Nos capítulos 1 e 9 ensinamos como criá-las.

0 - preto (ausência de cor)	3 - azul	6 - azul esverdeado
1 - verde	4 - vermelho	7 - carmim
2 - amarelo	5 - bege	8 - laranja

Usando a cor 0 no comando SET, ela não alterará a cor do ponto.

APÊNDICE B

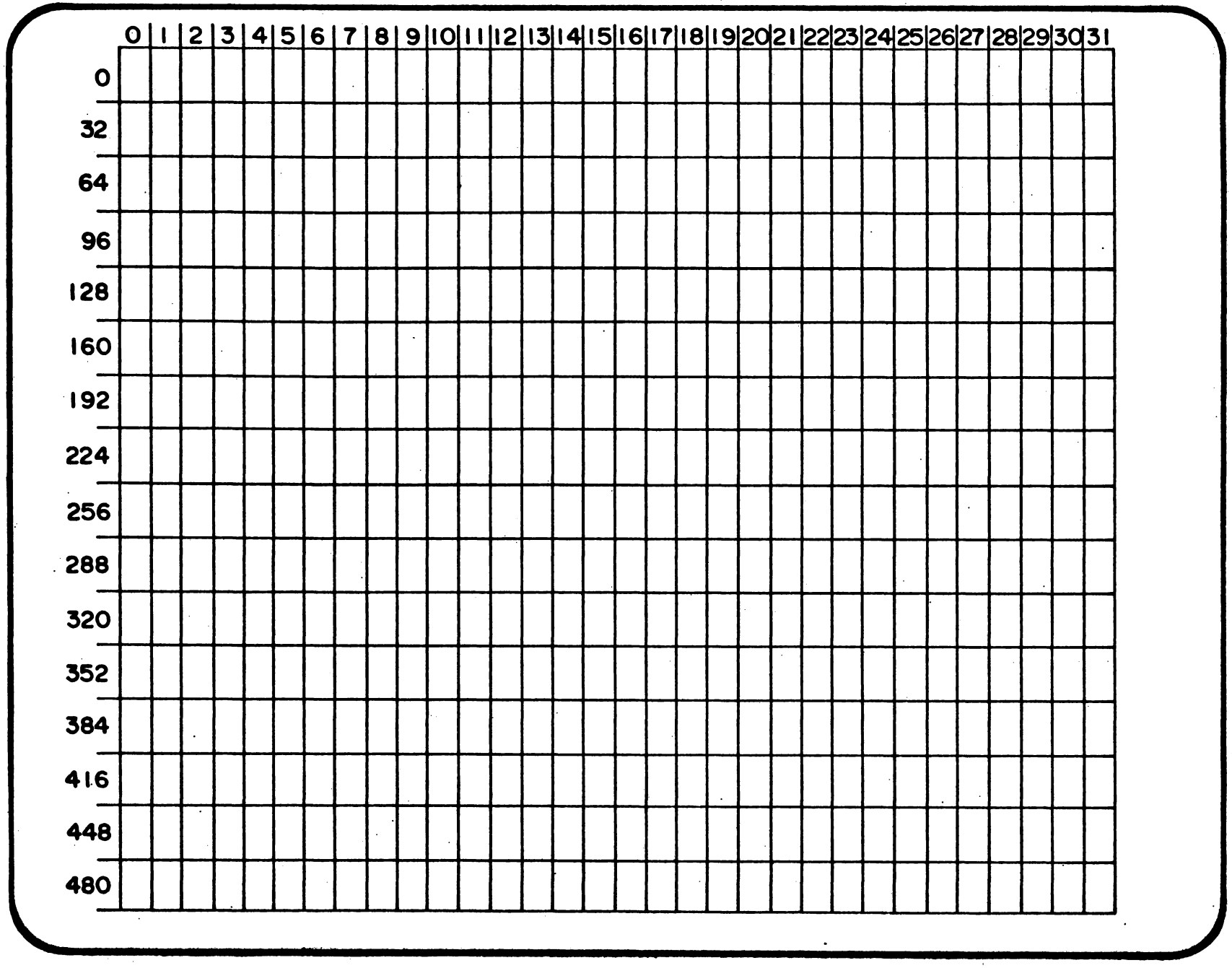
CARACTERES GRÁFICOS

A seguir apresentamos os códigos dos caracteres gráficos. Para produzi-los, use CHR$ com o código do caráter. Por exemplo, PRINT CHR$(129) mostra na tela o caráter 129.

Para criar estes caracteres em uma das cores abaixo, adicione o número apropriado ao código. Por exemplo, PRINT CHR$(129+16) mostra na tela o caráter 129 em amarelo (exceto a área verde).

+16 - amarelo
+32 - verde
+48 - vermelho
+64 - bege
+80 - azul esverdeado
+96 - carmim
+112 - laranja

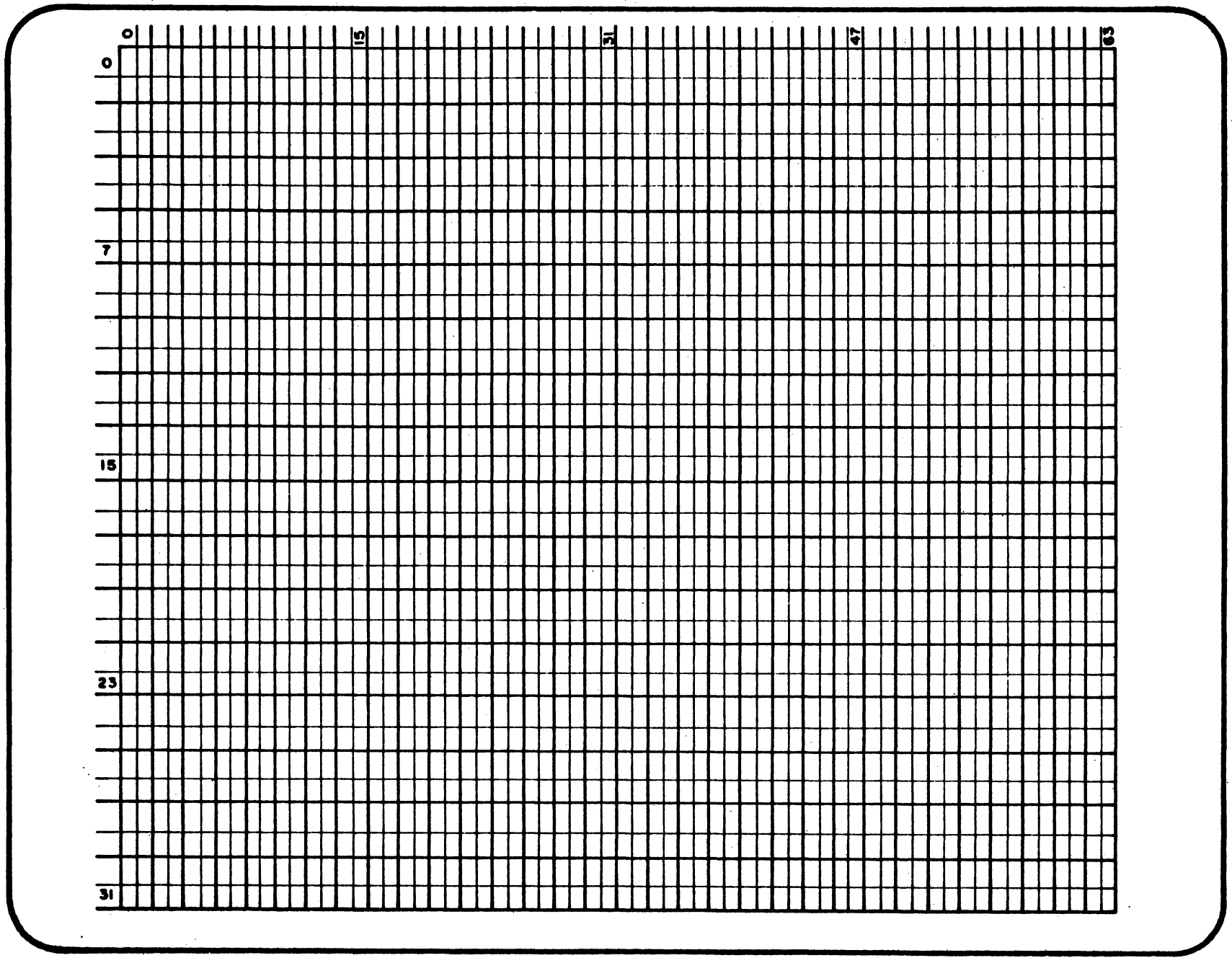
APÊNDICE C — POSIÇÕES PRINT @



APÊNDICE C

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255
256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287
288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319
320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351
352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383
384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415
416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447
448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479
480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511

APÊNDICE D — POSICIONAMENTO DE GRÁFICOS



APÊNDICE E

RELAÇÃO DE CÓDIGOS ASCII

A seguir, apresentamos a relação de códigos ASCII dos caracteres do teclado. A primeira coluna indica o caráter ; a segunda, representa o código em notação decimal (base 10); e a terceira é o código em notação hexadecimal (base 16).

CARATÉR	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL	CARÁTER	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL	CARÁTER	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL
ESPAÇO	32	20	4	52	34	H	72	48
!	33	21	5	53	35	I	73	49
"	34	22	6	54	36	J	74	4A
#	35	23	7	55	37	K	75	4B
$	36	24	8	56	38	L	76	4C
%	37	25	9	57	39	M	77	4D
&	38	26	:	58	3A	N	78	4E
'	39	27	;	59	3B	O	79	4F
(	40	28	<	60	3C	P	80	50
)	41	29	=	61	3D	Q	81	51
*	42	2A	>	62	3E	R	82	52
+	43	2B	?	63	3F	S	83	53
,	44	2C	@	64	40	T	84	54
-	45	2D	A	65	41	U	85	55
.	46	2E	B	66	42	V	86	56
/	47	2F	C	67	43	W	87	57
0	48	30	D	68	44	X	88	58
1	49	31	E	69	45	Y	89	59
2	50	32	F	70	46	Z	90	5A
3	51	33	G	71	47			

APÊNDICE E

RELAÇÃO DE CÓDIGOS DAS TECLAS ESPECIAIS

CARÁTER	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL	CARÁTER	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL
↑	94	5E	SHIFT ↑	95	5F
↓	10	0A	SHIFT ↓	91	5B
←	8	08	SHIFT ←	21	15
→	9	09	SHIFT →	93	5D
BREAK	3	03	SHIFT CLEAR	92	5C
CLEAR	12	0C			
ENTER	13	0D			

APÊNDICE E

RELAÇÃO DE CÓDIGOS DAS LETRAS MINÚSCULAS

A seguir apresentamos a relação de códigos ASCII das letras minúsculas. Para usar as letras minúsculas é preciso primeiro pressionar as teclas (SHIFT) (0) simultaneamente. As letras minúsculas aparecerão no vídeo em cores reversas (letras verdes e fundo preto).

CARÁTER	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL	CARÁTER	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL
a	97	61	n	110	6E
b	98	62	o	111	6F
c	99	63	p	112	70
d	100	64	q	113	71
e	101	65	r	114	72
f	102	66	s	115	73
g	103	67	t	116	74
h	104	68	u	117	75
i	105	69	v	118	76
j	106	6A	w	119	77
k	107	6B	x	120	78
l	108	6C	y	121	79
m	109	6D	z	122	7A

APÊNDICE F

. Erro no número do buffer. Você usou mais buffers que o computador reservou. Use o comando FILES para reservar mais buffers. Este erro também ocorre se você usar um número de buffer inválido (por exemplo buffer#-3) ou omitir o número do buffer na instrução FIELD (por exemplo FIELD 1 AS A$ em vez de FIELD#1, 1 AS A$). (Device Number)

DS Instrução direta. Existe uma instrução direta no arquivo. Este erro ocorre se você carregar um programa cujas instruções não tenham número de linha. (Direct Statement)

FC Chamada incorreta de função. Este erro ocorre quando você usa uma palavra BASIC com um parâmetro (número) fora dos limites admissíveis. Por exemplo SOUND (260, 260) e CLS(10) provocarão este erro. A instrução RIGHT$(S$,20) também provoca este erro se a variável S$ possuir menos que 20 caracteres. Existem outras situações em que ocorre este erro, como por exemplo, quando usamos vetores com índices negativos (A(-1)) ou quando chamamos funções USR sem termos fornecido (POKE) o seu endereço antes. (Funcion Call)

FD Formato de campo incorreto.Este erro ocorre quando você lê (INPUT) ou grava (PRINT) um dado usando um formato inadequado. Se tentar, por exemplo, ler um string com a instrução INPUT#1,A este erro ocorrerá (File Data).

FM Tipo de arquivo incorreto. O tipo de arquivo ("O", "I" ou "D") especificado na instrução OPEN não corresponde ao que se quer executar. Você está tentando, por exemplo, usar as instruções GET e PUT num arquivo tipo "I" ou "O" (use "D"). Este erro também ocorre se for usada a instrução WRITE num arquivo "I" (use "O") ou a instrução INPUT num arquivo "O" (use "I"). (File Mode)

APÊNDICE F

ID Instrução direta incorreta. Você só pode usar a instrução INPUT dentro de um programa. Não é possível usá-la como um comando (Illegal Direct statement)

IE Tentativa de ler um dado além do fim do arquivo. Use as instruções EOF e LOF para descobrir se você atingiu o fim de um arquivo. Quando atingir o fim de arquivo feche-o (CLOSE) (Input past End).

IO Erro de entrada ou saída. O computador está encontrando dificuldades para ler ou gravar informações num arquivo em fita. (Input/Output)

LS String muito longo. Um string pode ter no máximo 255 caracteres (String too Long).

NF Instrução NEXT sem FOR correspondente. Você usou uma instrução NEXT sem a instrução FOR correspondente. Este erro também ocorre se você inverter a ordem das instruções NEXT num ninho de FOR/NEXT (Next without For).

NO O arquivo não foi aberto. Você não pode ler ou gravar dados num arquivo sem antes abrí-lo (OPEN) (Not Open).

OD Não há dados nas instruções DATA para atender a todas as instruções READ. Verifique as instruções DATA e READ do seu programa (Out of Data).

OM Estouro de memória. Você já usou toda a memória disponível do computador. Tente alterar a estrutura do programa dividindo-o em vários módulos independentes e transformando cada módulo num programa autônomo (Out of Memory).

APÊNDICE F

OS Estouro de string. Não há espaço suficiente na memória do computador para executar as operações envolvendo strings. Use a instrução CLEAR no começo do programa para reservar mais espaço para os strings. (Out of String).

OV Estouro de variável numérica. Uma das operações aritméticas, no seu programa, gerou um número muito grande ou muito pequeno e o computador não tem condições de processar este número (OVerflow).

RG Instrução RETURN sem o GOSUB correspondente. Você usou uma instrução RETURN sem a instrução GOSUB correspondente. (Return without Gosub)

SN Erro de sintaxe. Este erro ocorre quando você escreve incorretamente uma palavra BASIC ou usa incorretamente a pontuação, os parênteses ou as aspas. Digite novamente a linha incorreta. (SiNtaxe)

ST Tentativa de usar strings em fórmulas muito complexas. Uma operação com strings está muito complexa para ser processada pelo computador. Desdobre esta operação em outras de menor porte. (STring)

TM Incompatibilidade de tipo de variável. Este erro ocorre quando você tenta atribuir um número a uma variável string (A$=3) ou um string a uma variável numérica (A="MICRO"). Este erro também ocorre se você não colocar o nome de um arquivo entre duas aspas (Type Mismatch)

UL Linha não definida. Existe no programa uma instrução GOTO ou GOSUB ou qualquer outra instrução de desvio pedindo ao computador que vá para uma linha cujo número não existe (Undefined Line).

APÊNDICE G

SUMÁRIO DO BASIC

COMANDO	DESCRIÇÃO	EXEMPLO
ABS	Retorna o valor absoluto da expressão fornecida.	PRINT ABS(-5*2)
ASC	Retorna o código ASCII do primeiro caráter do string fornecido.	PRINT ASC ("B") S = ASC (T$)
AUDIO	Liga ou desliga a saída do gravador cassete para o alto-falante da televisão.	AUDIO ON AUDIO OFF
CHR$	Retorna o caráter ASCII, de controle ou gráfico relativo ao código fornecido.	PRINT CHR$(67) Y$ = CHR$(92)
CLEAR	Reserva espaço de memória para armazenar string (sem o CLEAR, o computador reserva apenas 200 bytes). Se estiver carregando um programa em linguagem de máquina, você pode usar um segundo número para indicar o endereço mais alto do BASIC. Este comando também inicializa as variá-	CLEAR CLEAR 1000 CLEAR 200, 14000

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
	veis numéricas com zero e as variáveis string com nulos (LEN=0).	
CLOAD	Carrega um programa em BASIC do gravador cassete. Se não for indicado o nome do arquivo, é carregado o primeiro que for encontrado na fita. O nome do arquivo pode ter no máximo 8 caracteres	CLOAD CLOAD "PROG"
CLOADM	Carrega um programa em linguagem de máquina do gravador cassete. Você pode fornecer um número que será somado ao endereço de carregamento do progra - ma.	CLOADM "PROG" CLOADM CLOADM "PROG", 1000
CLOSE	Fecha os arquivos abertos. Cuidado, não se esqueça que este comando esva - zia o buffer de comunicação.	CLOSE#-1 CLOSE#-2 CLOSE
CLS	Limpa a tela e coloca a cor do código fornecido. Caso não seja indicado ne nhum código, é usado a cor verde.	CLS 3 CLS
CONT	Continua a execução de um programa interrompido por uma instrução STOP ou (BREAK)	CONT

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
CSAVE	Salva um programa na fita cassete (o nome do programa pode ter no máximo oito caracteres)... Se for especificado um A após o nome, ele será salvo no formato ASCII.	CSAVE CSAVE "PROG" CSAVE "PROG", A
DATA	Armazena dados dentro do programa. Use a instrução READ para ler estes dados.	DATA 5, 3, PAPEL DATA PERA, UVA
DIM	Reserva espaço na memória para vetores e matrizes numéricas ou strings.	DIM R(12), T(14,27) DIM X$(5), Y$(7,9)
END	Termina o programa. Não é possível usar o comando CONT após uma instrução END.	IF A = 1 THEN END END
EOF	Verifica se foi atingido o fim de um arquivo. Se EOF=0 ainda existem dados a serem lidos. Se EOF=1 não há mais dados. -1 - gravador cassete 0 - teclado	IF EOF(-1)<>0 THEN CLOSE IF EOF(-1) = 0 THEN INPUT #-1, A$
EXEC	Transfere o controle para um programa em linguagem de máquina começando no endereço especificado. Se o endereço for omitido, o controle é	EXEC EXEC 32453

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
	transferido para o endereço indicado no último comando CLOADM.	
FOR...TO...STEP/NEXT	Cria um loop no programa que o computador repete do primeiro ao último número especificado. Use STEP para indicar de quanto é o incremento a cada passagem pelo NEXT. Se não for fornecido o STEP, é usando o número 1.	FOR X=2 TO 5 / NEXT X FOR A=1 TO 10 STEP 5/NEXT A FOR M=30 TO 15 STEP 5/NEXT M
GOSUB	Transfere o controle para uma rotina BASIC começando na linha indicada.	GOSUB 500 GOSUB 1000
GOTO	Desvia a execução do programa para a linha indicada.	GOTO 10 GOTO 1110
IF teste THEN ação 1 ELSE ação 2	Faz o teste. Se o resultado for verdadeiro, o computador executa a ação 1. Caso contrário (falso), executa a ação 2.	IF A=5 THEN 30 IF B$="BRASIL" THEN PRINT "CORRETO" ELSE PRINT "ERRADO"
INKEY$	Testa o teclado e fornece o código ASCII da tecla pressionada (caso tenha sido pressionada alguma).	A$ = INKEY$

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
INPUT	Interrompe a execução do programa e faz o computador esperar a entrada de dados pelo teclado.	INPUT X$ INPUT "NOME"; X$ INPUT A, B$
INT	Retorna a parte inteira da expressão fornecida.	X = INT(A*5.2)
JOYSTK	Retorna a coordenada horizontal ou vertical do joystick esquerdo ou direito. 0 - Coord. horizontal do joystick direito 1 - Coord. vertical do joystick direito 2 - Coord. horizontal do joystick esquerdo 3 - Coord. vertical do joystick esquerdo	X = JOYSTK(0) Y = JOYSTK(1)
LEFT$	Retorna os n primeiros caracteres (contando a partir da esquerda) do string fornecido.	P$ = LEFT$(M$,7)
LEN	Retorna o comprimento em bytes, do string fornecido.	X = LEN(X$)
LIST	Mostra todo ou parte do programa na tela.	LIST LIST 10-50

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
		LIST -1000 LIST 1000- LIST 20
LLIST	Imprime todo ou parte do programa na impressora	LLIST LLIST 10-100
MEM	Indica qual o tamanho do espaço que ainda permanece livre na memória do computador.	PRINT MEM
MID$	Retorna um substring do string fornecido começando na posição indicada com um comprimento determinado.	PRINT MID$ (A$, 3, 7)
MOTOR	Liga ou desliga o gravador cassete.	MOTOR ON MOTOR OFF
NEW	Apaga o conteúdo da memória.	NEW
ON... GOSUB...	Transfere o controle para uma das subrotinas especificadas dependendo do valor da expressão.	ON Y*A GOSUB 1, 5, 7

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
ON... GOTO...	Desvia a execução do programa para uma das linhas indicadas dependendo do valor da expressão.	ON X+2 GOTO 5, 7, 9
OPEN	Abre um arquivo da seguinte forma: #0 - Tela ou teclado #-1 - Gravador cassete #-2 - Impressora A abertura pode ser: I - entrada O - saída Obs.: Não é preciso abrir nenhum arquivo para ler dados do teclado ou mostrar informações na tela.	OPEN "I",# -1, "ARQ" OPEN "O", # -1, "DADOS"
PEEK	Retorna o conteúdo do endereço de memória especificado.	A = PEEK(32706)
POINT	Testa se o ponto indicado está ativado ou desativado. O valor retornado pode ser: -1 - O ponto está em modo texto 0 - O ponto está desativado Cod - Código da cor que o ponto possui	IF POINT(15,12)=3 THEN PRINT "AZUL"

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
POKE	Coloca um código (0-255) no endereço de memória especificado.	POKE 15872, 123
PRINT	Imprime a lista de dados fornecida no periférico indicado. Se você não indicar o periférico, os dados serão mostrados na tela. Consulte o comando OPEN para conhecer os códigos dos periféricos.	PRINT "COLOR" PRINT B$ PRINT # -1, A$;B$ PRINT # -2, A$;"COLOR"
PRINT @	Imprime os dados na tela, a partir da posição especificada.	PRINT @ 194,"01"; A$
READ	Lê o próximo item de uma instrução DATA e atribui o dado a uma variável especificada.	READ A$ READ A, B$, C
REM	Permite a colocação de comentários num programa. Tudo que for digitado após o REM (antes de pressionar (ENTER) é ignorado pelo computador.)	REM ISTO E' UM COMENTARIO
RESET	Desativa o ponto ativado pelo comando SET	RESET (14,15)
RESTORE	Retorna o indicador da instrução DATA para a sua posição inicial.	RESTORE
RETURN	Transfere o controle do programa para a linha imediatamente abaixo do	RETURN

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
	último GOSUB executado	
RIGHT$	Retorna os n primeiros caracteres (contando a partir da direita) do string fornecido.	Z$ = RIGHT$(A$,5)
RND	Retorna um número inteiro aleatório entre 1 e o número indicado.	A = RND(15)
RUN	Executa um programa.	RUN
SET	Ativa a posição especificada da tela na cor indicada.	SET(14,13,3)
SGN	Retorna o sinal da expressão numérica fornecida -1 - se a expressão for negativa 0 - se a expressão for zero 1 - se a expressão for positiva	A = SGN(-4*B) B = SGN(-5) B = SGN(438.37)
SIN	Retorna o seno do ângulo dado em radianos.	Y = SIN(5*B/A)
SKIPF	Pula para o próximo programa do cassete ou para o fim do programa indicado.	SKIPF SKIPF "PGM"

APÊNDICE G

COMANDO	DESCRIÇÃO	EXEMPLO
SOUND	Soa um determinado tom com uma determinada duração.	SOUND 128,3
STOP	Interrompe a execução de um programa.	STOP
STR$	Converte uma expressão numérica em string.	S$ = STR$(A*B)
TAB	Inicia a impressão dos dados a partir da coluna indicada.	PRINT TAB(2) "COLOR" PRINT # -2, TAB(5); A$
USR	Chama uma rotina em linguagem de máquina cujo endereço está armazenado nas posições 275 e 276.	X = USR(Y)
VAL	Converte um string em um número.	A = VAL(B$)

APÊNDICE H

CARACTERES DO TECLADO

(←)	- Apaga o último caráter digitado; move o cursor uma posição para trás. O cursor é o ponto que fica pis - cando na tela.
(SHIFT) (←)	- Volta o cursor para o início da linha atual, apagando tudo que foi digitado.
(BREAK)	- Interrompe a execução do programa.
(CLEAR)	- Limpa a tela.
(ENTER)	- Indica o fim da linha atual.
(ESPACO)	- Gera um caráter em branco e move o cursor uma posição para frente.
(SHIFT) (@)	- Provoca uma pausa na execução do programa. Pressione qualquer tecla para continuar.
(SHIFT) (0)	- Permite a passagem do modo letras maiúsculas para o modo letras minúsculas e vice-versa. As letras minúsculas são representadas por letras maiúsculas em cores reversas.

SÍMBOLOS BASIC

" "	- Indica que a informação entre aspas é uma constante (string).
:	- Separa os comandos escritos numa mesma linha.
( )	- Indica ao computador que as operações entre parênteses devem ser executadas primeiro.
;	- Faz com que as constantes e variáveis sejam impressas coladas umas nas outras.

APÊNDICE H

OPERADORES BASIC

+	combina strings (concatenação)
+	adição
-	subtração
*	multiplicação
/	divisão
=	igualdade
>	maior que
> = ou = >	maior que ou igual a
< = ou = <	menor que ou igual a
<	menor que
< > ou > <	diferente de
AND	conjunção E
OR	conjunção OU
NOT	advérbio NÃO

APÊNDICE I

RESPOSTAS DOS EXERCÍCIOS

CAPÍTULO 4

Tons obtidos da base da seqüência para o alto e depois do alto para a base.

```
10 FOR X=1 TO 255
20 SOUND X,1
30 NEXT X
40 FOR X=255 TO 1 STEP -1
50 SOUND X,1
60 NEXT X
```

CAPÍTULO 5

Linhas a acrescentar ao programa de relógio.

```
92 FOR T=200 TO 210 STEP 5
94 SOUND T,1
95 NEXT T
97 FOR T=210 TO 200 STEP -5
98 SOUND T,1
99 NEXT T
```

Programa que mostra 9 cores. Cada cor permanece na tela durante 1 segundo.

```
10 FOR C=0 TO 8
20 CLS(C)
30 FOR X=1 TO 460
40 NEXT X
50 NEXT C
```

CAPÍTULO 7

Jogo "craps".

```
10  CLS
20  A=RND(6)
30  B=RND(6)
40  R=A+B
50  PRINT@ 204,A
60  PRINT@ 208,B
70  PRINT@ 232,"VOCE JOGOU UM"R
80  IF R=2 THEN 500
90  IF R=3 THEN 500
100 IF R=12 THEN 500
```

APÊNDICE I

```
110 IF R=7 THEN 600
120 IF R=11 THEN 600
130 FOR X=1 TO 800
140 NEXT X
150 CLS
160 PRINT@ 227,"JOGUE OUTRO"R"E
VOCE VENCE"
170 PRINT@ 259,"JOGUE UM 7 E VOC
E PERDE"
180 PRINT@ 416,"PRESSIONE <ENTER
> QUANDO ESTIVER"
185 PRINT@ 448,"PRONTO PARA SUA
PROXIMA JOGADA"
190 INPUT A$
200 X=RND(6)
210 Y=RND(6)
220 Z=X+Y
225 CLS
230 PRINT@ 204,X
240 PRINT@ 208,Y
250 PRINT@ 232,"VOCE JOGOU UM"Z
260 IF Z=R THEN 500
270 IF Z=7 THEN 600
280 GOTO 180
500 FOR X=1 TO 1000
510 NEXT X
515 CLS
520 PRINT@ 232,"VOCE E' O VENCED
OR"
530 PRINT@ 299,"PARABENS!!!!"
540 GOTO 630
600 FOR X=1 TO 1000
610 NEXT X
615 CLS
620 PRINT@ 260,"SINTO MUITO, VOC
E PERDEU"
630 PRINT@ 457,"O JOGO ACABOU"
```

Programa da Roleta Russa.

```
5 FOR N=1 TO 10
10 PRINT "ESCOLHA UMA CAMARA (1-
10)"
20 INPUT X
30 IF X=RND(10) THEN 100
40 SOUND 200,1
50 PRINT "--CLICK--"
60 NEXT N
65 CLS
70 PRINT@ 234,"PARABENS!!!"
80 PRINT@ 293,"VOCE LUTOU PARA V
IVER"
90 END
100 FOR T=133 TO 1 STEP -5
110 PRINT "BANG!!!!!!"
120 SOUND T,1
130 NEXT T
140 CLS
150 PRINT@ 225,"SINTO MUITO, VOC
E ESTA' MORTO"
160 SOUND 1,50
170 PRINT@ 291,"PROXIMA VITIMA,
POR FAVOR"
```

APÊNDICE I

CAPÍTULO 10

Teste seus conhecimentos matemáticos.

```
 5   CLS
 6   PRINT@ 227,"SEU NOME (6 CARA
C)";
 8   INPUT N$
 10  CLS
 15  T=T+1
 20  X=RND(100)
 30  Y=RND(100)
 40 PRINT@ 228,"QUANTO E'"X"+"Y;
 45  INPUT A
 50  IF A=X+Y THEN 82
 60 PRINT@ 328,"A RESPOSTA E'"X+Y
 70 PRINT@ 352,"ESPERAMOS QUE  TE
NHA MAIS  SORTE"
 75 PRINT@ 384,"DA PROXIMA VEZ,"N
$
 80  GOTO 100
 82  CLS(7)
 83  FOR M=1 TO 4
 84  SOUND 175,1
 85  SOUND 200,1
 86  NEXT M
 87  CLS
 90  PRINT@ 231,"CORRETO,"N$"!!!"
 95  C=C+1
 97 PRINT@ 294,"ESTA E' A"C"RESPO
STA"
 98 PRINT@ 325,"CORRETA EM"T"PERG
UNTAS"
 99  PRINT@ 359,C/T*100"% DE ACER
TO"
 100 PRINT@ 416,"PRESSIONE <ENTER
> QUANDO ESTIVER"
 102 PRINT@ 448,"PRONTO PARA OUTR
O TESTE"
 105 INPUT A$
 110 GOTO 10
```

CAPÍTULO 11

Tabela de quadrados

```
 5     CLS
 7     PRINT@ 38,"TABELA DE QUADRA
DOS"
 8     PRINT
 10    P=2
 20    FOR N=2 TO 10
 25    GOSUB 2000
 30    PRINT N"*"N"="E,
 40    NEXT N
 50    END
 2000 REM FORMULA PARA ELEVAR UM
 2001 REM NUMERO A UMA POTENCIA
 2010 E=1
 2020 FOR X=1 TO P
 2030 E=E*N
 2040 NEXT X
 2050 IF P=0 THEN E=1
 2060 RETURN
```

APÊNDICE I

CAPÍTULO 12

Editando uma sentença.

```
10 PRINT"DIGITE UMA SENTENCA:"
15 INPUT S$
20 PRINT"DIGITE UMA FRASE DA SEN
TENCA PARA SER ELIMINADA:"
23 INPUT D$
25 L=LEN(D$)
30 PRINT"DIGITE UMA SENTENCA PAR
A SUBSTITUIR A FRASE ELIMINADA:"
35 INPUT R$
40 FOR X=1 TO LEN(S$)
50 IF MID$(S$,X,L)=D$ THEN 100
60 NEXT X
70 PRINT D$",NAO ESTA' NA SUA SE
NTENCA"
80 GOTO 20
100 E=X-1+LEN(D$)
110 NS$=LEFT$(S$,X-1)+R$+RIGHT$(
S$,LEN(S$)-E)
120 PRINT"A NOVA SENTENCA E':"
130 PRINT NS$
```

CAPÍTULO 13

Teste de datilografia.

```
10 CLS
20 INPUT"PRESSIONE <ENTER> QUAND
O ESTIVERPRONTO PARA DATILOGRA
FAR ESTASENTENCA";E$
30 PRINT"AGORA E' TEMPO PARA TOD
O SER BOM"
40 T=1
50 A$=INKEY$
60 IF A$="" THEN 100
70 PRINT A$;
80 B$=B$+A$
90 IF LEN(B$)=32 THEN 120
100 T=T+1
110 GOTO 50
120 S=T/74
130 M=S/68
140 R=8/M
142 FOR X=1 TO 32
144 IF MID$("AGORA E' TEMPO PARA
 TODO SER BOM",X,1)<>MID$(B$,X,1
) THEN E=E+1
146 NEXT X
150 PRINT
160 PRINT"VOCE DATILOGRAFOU"R"LE
TRAS POR MINUTO"
170 PRINT"COM"E"ERROS"
```

APÊNDICE I

CAPÍTULO 15

Controle do cursor

```
10 CLS(0)
20 H=63
25 SET(H,14,3)
30 A$=INKEY$
40 IF A$=CHR$(8) THEN 60
45 IF A$=CHR$(9) THEN 100
50 GOTO 30
60 H=H-1
65 IF H<0 THEN H=0: GOTO 30
70 SET(H,14,3)
75 RESET(H+1,14)
80 GOTO 30
100 H=H+1
110 IF H>63 THEN H=63: GOTO 30
120 SET(H,14,3)
130 RESET(H-1,14)
140 GOTO 30
```

CAPÍTULO 21

Processador de palavras

```
1 CLEAR 1000
5 DIM A$(50)
7 CLS
10 PRINT "ESCREVA UM PARAGRAFO"
16 REM
20 PRINT "PRESSIONE </> QUANDO T
ERMINAR"
30 X=1
40 A$=INKEY$
50 IF A$="" THEN 40
60 PRINT A$;
70 IF A$="/" THEN 105
80 A$(X)=A$(X)+A$
90 IF A$="." OR A$="?" OR A$="!"
 THEN X=X+1
100 GOTO 40
105 PRINT : PRINT
110 INPUT "(1) IMPRESSAO OU (2)
REVISAO"; R
120 CLS
130 ON R GOSUB 1000, 2000
140 GOTO 105
1000 REM IMPRESSAO DO PARAGRAFO
1010 FOR Y=1 TO X-1
1020 PRINT A$(Y);
1030 NEXT Y
1040 RETURN
2000 REM REVISAO DO PARAGRAFO
2010 FOR Y=1 TO X-1
2020 PRINT Y "--" A$(Y)
```

APÊNDICE I

```
2030 NEXT Y
2040 INPUT "SENTENCA A SER REVIS
TA"; S
2045 IF S>X-1 OR S<1 THEN 2040
2050 PRINT A$(S)
2060 PRINT "FORNECA A SENTENCA A
 SER ELIMINADA"
2070 INPUT D$
2080 L=LEN(D$)
2090 PRINT "FORNECA A NOVA SENTE
NCA"
2100 INPUT R$
2110 FOR Z=1 TO LEN(A$(S))
2120 IF MID$(A$(S),Z,L)=D$ THEN
 2160
2130 NEXT Z
2140 PRINT D$ "-- NAO EXISTE NA
SENTENCA"
2150 GOTO 2060
2160 E=Z-1+LEN(D$)
2170 A$(S)=LEFT$(A$(S),Z-1) + R$
 + RIGHT$(A$(S),LEN(A$(S))-E)
2180 RETURN
```

CAPÍTULO 23

Ordenação alfabética de uma coleção de livros.

```
1 CLS: CLEAR 1000: DIM T$(100),
A$(100), S$(100), M$(100), Z(100
)
2 PRINT "POSICIONE A FITA E PRES
SIONE PLAY E RECORD";R$
4 INPUT "PRESSIONE <ENTER> QUAND
O ESTIVERPRONTO"; R$
8 REM
9 REM SAIDA PARA FITA
10 OPEN "O", #-1, "LIVROS"
15 CLS: PRINT "FORNECA OS TITULO
S E DIGITE <XX>QUANDO TERMINAR"
20 INPUT "TITULO";T$
25 IF T$="XX" THEN 50
26 INPUT "AUTOR";A$
28 INPUT "ASSUNTO";S$
30 PRINT #-1, T$, A$, S$
40 GOTO 15
50 CLOSE #-1
60 CLS: PRINT "REBOBINE A FITA E
 PRESSIONE PLAY"
70 INPUT "PRESSIONE <ENTER> QUAN
DO ESTIVERPRONTO"; R$
74 REM
76 REM ENTRADA DE DADOS A PARTIR
 DA FITA
78 B=1
80 OPEN "I", #-1, "LIVROS"
85 IF EOF(-1) THEN 120
90 INPUT #-1, T$(B), A$(B), S$(B
)
```

APÊNDICE I

```
95 B=B+1
110 GOTO 85
120 CLOSE #-1
490 PRINT
500 INPUT "ORDENACAO POR (1)-TIT
ULO (2)-AUTOR (3)-ASSUNTO";A
510 IF A>3 OR A<1 THEN 500
520 ON A GOSUB 1000, 2000, 3000
530 GOSUB 4000
540 PRINT
550 FOR X=1 TO B-1
560 PRINT "TITULO  :";T$(Z(X))
570 PRINT "AUTOR   :";A$(Z(X))
580 PRINT "ASSUNTO :";S$(Z(X))
590 NEXT X
600 PRINT: GOTO 500
800 REM
900 REM CONSTRUCAO DO VETOR M$
1000 FOR X=1 TO B-1
1010 M$(X)=T$(X)
1020 NEXT X
1030 RETURN
2000 FOR X=1 TO B-1
2010 M$(X)=A$(X)
2020 NEXT X
2030 RETURN
3000 FOR X=1 TO B-1
3010 M$(X)=S$(X)
3020 NEXT X
3030 RETURN
3900 REM
```

```
4000 REM ROTINA DE ORDENACAO (SO
RT)
4005 T=1
4010 X=0
4020 X=X+1
4030 IF X>B-1 THEN RETURN
4040 IF M$(X)="ZZ" THEN 4020
4050 FOR Y=1 TO B-1
4060 IF M$(Y) < M$(X) THEN X=Y
4065 Z(T)=X
4080 NEXT Y
4085 T=T+1
4090 M$(X)="ZZ"
4100 GOTO 4010
```

APÊNDICE I

CAPÍTULO 24

Jogando com um baralho "Bi-dimensional".

```
10 DIM S$(4), N$(13), T(4,13)
20 DATA ESPADAS,COPAS,OUROS,PAUS
30 FOR X=1 TO 4
40 READ S$(X)
50 NEXT X
60 DATA AS,2,3,4,5,6,7,8,9,10,VA
LETE,DAMA,REI
70 FOR X=1 TO 13
80 READ N$(X)
90 NEXT X
100 FOR S=1 TO 4
110 FOR N=1 TO 13
120 T(S,N)=(S-1)*13+N
130 NEXT N,S
140 FOR X=1 TO 52
150 S=RND(4): N=RND(13)
160 IF T(S,N)=0 THEN 150
170 T(S,N)=0
180 PRINT N$(N)"-"S$(S),
190 NEXT
```

APÊNDICE J

SUBROTINAS CIENTÍFICAS

Estas subrotinas permitem executar programas que utilizam funções matemáticas avançadas e que não estão diretamente disponíveis em COLOR BASIC.

Cada listagem de subrotina tem um conjunto de instruções ao lado. Estude-as atentamente, você verá que algumas delas fazem uso de outras para cálculos internos. Você terá que digitar estas "subrotinas auxiliares" sempre que as instruções assim o exigirem.

NOTA: A precisão das subrotinas é menor que a das funções e operadores matemáticos do COLOR BASIC. Isto se deve a dois fatos:

1 - As subrotinas contêm muitos cálculos encadeados, que tendem a aumentar os pequenos erros de operações isoladas.

2 - Estas subrotinas são apenas aproximações das funções que elas substituem. Em geral, as subrotinas têm precisão até a 5ª. ou 6ª. casa decimal para a maior parte dos valores permitidos e conforme este valor se aproxima dos limites inferior ou superior da faixa permitida, ocorre uma diminuição da precisão.

APÊNDICE J

RAIZ QUADRADA

Determina : SQR(X), $\sqrt{X}$
Entrada : X, deve ser maior ou igual a zero
Saída : Y
Variáveis usadas : W, Z(internamente)
Subrotinas usadas: nenhuma
Como chamar : GOSUB 30030

```
30000 END
30010 REM *RAIZ QUADRADA* ENTRADA X, SAIDA Y
30020 REM VARIAVEIS USADAS: W&Z INTERNAMENTE
30030 IF X=0 THEN Y=0: RETURN
30040 IF X>0 THEN 30060
30050 PRINT "RAIZ DE NUMERO NEGATIVO?": STOP
30060 Y = X * .5 : Z=0
30070 W = (X/Y - Y) * .5
30080 IF (W=0) + (W=Z) THEN RETURN
30090 Y=Y+W : Z=W : GOTO 30070
```

EXPONENCIAÇÃO

Determina : X^Y (X na potência Y)
Entrada : X, Y. Se X for menor que zero, Y deve ser um número inteiro ímpar.
Saída : P
Variáveis usadas : E, L, A, B, C (internamente). O valor de X é alterado.
Subrotinas usadas: LOG e EXPONENCIAL
Como chamar : GOSUB 30120

```
30000 END
30100 REM *EXPONENCIACAO* ENTRADA X,Y: SAIDA P
30110 REM VARIAVEIS USADAS E,L,A,B,C INTERNAMENTE
30120 P=1 : E=0 : IF Y=0 THEN RETURN
30130 IF (X<0)AND(INT(Y)=Y) THEN P=1-2*Y+4*INT(Y/2
): X=-X
30140 IF X<>0 THEN GOSUB 30190: X=Y*L: GOSUB 30250
30150 P=P*E: RETURN
```

APÊNDICE J

LOGARÍTMO NATURAL E COMUM

Determina : LOG(X) na base e, e LOG(X) na base 10
Entrada : X maior ou igual a zero
Saída : L é o logarítmo natural, X é o logarítmo decimal
Variáveis usadas : A, B, C (internamente). O valor de X é alterado
Subrotinas usadas: nenhuma
Como chamar : GOSUB 30190

```
30000 END
30170 REM *LOG NATURAL E COMUM: ENTRADA X, SAIDA L
,X
30180 REM L E O LOG NATURAL, X E O LOG DECIMAL
30190 E=0: IF X<0 THEN PRINT "LOG INDEFINIDO PARA"
;X: STOP
30195 A=1: B=2: C=.5
30200 IF X>=A THEN X=C*X: E=E+A: GOTO 30200
30205 IF X<C THEN X=B*X: E=E-A: GOTO 30205
30210 X=(X-.707107)/(X+.707107): L=X*X
30215 L=(((.598979*L+.961471)*L+2.88539)*X+E-.5)*.
693147
30220 IF ABS(L)<1E-6 THEN L=0
30225 X=L*.4342945: RETURN
```

EXPONENCIAL

Determina : EXP(X) (e na potência X)
Entrada : X
Saída : E
Variáveis usadas : L, A (internamente). O valor de X é alterado
Subrotinas usadas: nenhuma
Como chamar : GOSUB 30250

```
30000 END
30240 REM *EXPONENCIAL* ENTRADA X, SAIDA E
30245 REM VARIAVEIS USADAS L,A INTERNAMENTE
30250 L=INT(1.4427*X)+1 : IF L<127 THEN 30265
30255 IF X>0 THEN PRINT "LIMITE ULTRAPASSADO":STOP
30260 E=0: RETURN
30265 E=.693147*L-X: A=1.32988E-3-1.41316E-4*E
30270 A=((A*E-8.30136E-3)*E+4.16574E-2)*E
30275 E=(((A-.166665)*E+.5)*E-1)*E+1: A=2
30280 IF L<=0 THEN A=.5: L=-L: IF L=0 THEN RETURN
30285 FOR X=1 TO L: E=A*E: NEXT X: RETURN
```

APÊNDICE J

TANGENTE

Determina : TAN(X)
Entrada : X em graus
Saída : Y
Variáveis usadas : nenhuma
Subrotinas usadas: COSSENO
Como chamar : GOSUB 30310

```
30000 END
30300 REM *TANGENTE* ENTRADA X EM GRAUS, SAIDA Y
30310 IF ABS(SIN((90-X)/57.29577951))<1E-7 THEN PR
INT "INDEFINIDO": STOP
30320 Y=SIN(X/57.29577951)/SIN((90-X)/57.29577951)
30330 RETURN
```

COSSENO

Determina : COS(X)
Entrada : X em graus
Saída : Y
Variáveis usadas : nenhuma
Subrotinas usadas: nenhuma
Como chamar : GOSUB 30360

```
30000 END
30350 REM *COSSENO* ENTRADA X EM GRAUS, SAIDA Y
30360 Y=SIN((90-X)/57.29577951)
30365 RETURN
```

APÊNDICE J

ARCO COSSENO

Determina : ARCCOS(S), ângulo cujo cosseno é S
Entrada : S, 0 < =S < =1
Saída : Y em graus e W em radianos
Variáveis usadas : X, Z (internamente)
Subrotinas usadas: ARCO SENO
Como chamar : GOSUB 30500

```
30000 END
30500 REM *ARCO COSSENO* ENTRADA S, SAIDA Y,W
30510 REM Y EM GRAUS, W EM RADIANOS
30520 GOSUB 30550: Y=90-Y: W=1.570796-W: RETURN
```

ARCO SENO

Determina : ARCSEN(S); ângulo cujo seno é S
Entrada : S, 0 < =S < =1
Saída : Y em graus e W em radianos
Variáveis usadas : X, Y (internamente)
Subrotinas usadas: nenhuma
Como chamar : GOSUB 30550

```
30000 END
30530 REM *ARCO SENO* ENTRADA S, SAIDA Y,W
30535 REM Y EM GRAUS, W EM RADIANOS
30540 REM VARIAVEIS USADAS X,Z
30550 X=S: IF ABS(S)<=.707107 THEN 30610
30560 X=1-S*S: IF X<0 THEN PRINT S;"FORA DO LIMITE
": STOP
30565 IF X=0 THEN W=90/57.29577951: GOTO 30630
30570 W=X/2 : Z=0
30580 Y=(X/W-W)/2: IF (ABS(Y)<.1E-8)AND(Y=Z) THEN
X=W: GOTO 30610
30600 W=W+Y: Z=Y: GOTO 30580
30610 Y=X+X*X*X/6+X*X*X*X*X*.075+X*X*X*X*X*X*X*4.4
64286E-2
30620 W=Y+X*X*X*X*X*X*X*X*X*3.038194E-2
30625 IF ABS(S)>.707107 THEN W=1.570796-W
30630 Y=W*57.29577951: RETURN
```

APÊNDICE J

ARCO TANGENTE

Determina	: ATN(X), ângulo cuja tangente é X
Entrada	: X
Saída	: C em graus e A em radianos
Variáveis usadas	: B, T (internamente). O valor de X é alterado
Subrotinas usadas	: nenhuma
Como chamar	: GOSUB 30690

```
30000 END
30660 REM *ARCO TANGENTE* ENTRADA X, SAIDA C,A
30670 REM C EM GRAUS, A EM RADIANOS
30680 REM VARIAVEIS USADAS B,T
30690 T=SGN(X): X=ABS(X): C=0
30700 IF X>1 THEN C=1: X=1/X
30710 A=X*X
30720 B=((2.86623E-3*A-1.61657E-2)*A+4.29096E-2)*A
30730 B=((((B-7.5289E-2)*A+.106563)*A-.142089)*A+.
199936)*A
30740 A=((B-.333332)*A+1)*X
30750 IF C=1 THEN A=1.570796-A
30760 A=T*A: C=A*57.29577951: RETURN
```

APÊNDICE K

PROGRAMAS SIMPLES

Arma espacial

```
10 CLEAR 1000
20 FOR Y=0 TO 1
30 C=(Y+1)*16
40 S$(Y)=CHR$(131+C)+CHR$(139+C)
+CHR$(130+C)
50 S2$(Y)=CHR$(128+C)+CHR$(136+C
)
60 NEXT Y
100 FOR Y=0 TO 1
105 C=JOYSTK(0)
110 A(Y)=JOYSTK(0+Y*2)
120 B(Y)=JOYSTK(1+(Y*2))
130 IF A(Y)>59 THEN A(Y)=59
140 B(Y)=INT(B(Y)/4)*4
150 L(Y)=B(Y)*8+INT(A(Y)/2)
160 IF L(Y)>=480 THEN L(Y)=L(Y)-
32
170 NEXT Y
180 CLS(0)
190 FOR Y=0 TO 1
200 PRINT @L(Y), S$(Y);
210 PRINT @L(Y)+32, S2$(Y);
220 NEXT Y
500 P=PEEK(65280)
510 IF P=125 OR P=253 THEN GOSUB
 1000
530 GOTO 100
800 REM
900 REM ROTINA DE TIRO
1000 V1=INT(B(1)/2)+1
1010 H1=A(1)+2
1020 IF A(1) > A(0) THEN 1100
1030 FOR H=H1+3 TO 63
1040 IF POINT(H,V1)=2 THEN SOUND
 100,2
1050 SET(H,V1,4)
1060 IF H <= H1+4 THEN 1080
1070 RESET(H-2,V1)
1080 NEXT H
1090 RETURN
1100 FOR H=H1 TO 4 STEP -1
1110 IF H=H1 THEN 1160
1120 IF POINT(H-4,V1)=2 THEN SOU
ND 100,2
1130 SET(H-4,V1,4)
1140 IF H >= H1-2 THEN 1160
1150 RESET(H-2,V1)
1160 NEXT H
1170 RETURN
```

Bola quicando

```
 5 CLEAR 12
 8 INPUT "COR DO FUNDO (1-8)";C
 9 CLS(C)
 10 X=13: Y=13
 15 XM=20: YM=15
 400 F=0
 410 XT=X: YT=Y
 420 X=X+XM: Y=Y+YM
 430 TX=X: TY=Y: T1=XM: T2=YM
 440 GOSUB 1000
 450 X=TX: Y=TY: XM=T1: YM=T2
 455 H=INT(XT/2)*2: V=INT(VT/2)*2
 460 SET(H,V,C): SET(H+1,V,C)
 462 SET(H,V+1,C): SET(H+1,V+1,C)
 470 RESET(X,Y)
 480 GOTO 400
 499 REM
 1000 REM VERIFICACAO DOS LIMITES
 1010 IF TX>63 THEN TX=63: T1=-T1
 1020 IF TX< 0 THEN TX= 0: T1=-T1
 1030 IF TY>31 THEN TY=31: T2=-T2
 1040 IF TY< 0 THEN TY= 0: T2=-T2
 1099 RETURN
```

APÊNDICE K

Blackjack

```
5 REM CONSTRUCAO DOS VETORES
7 DIM S$(5), N$(13), D(52), P(5)
, C(5)
10 DATA 16, 32, 48, 96, 1
20 DATA **AS**, *DOIS*, *TRES*,
QUATRO, CINCO*, *SEIS*, *SETE*,
*OITO*, *NOVE*, *DEZ**, VALETE,
*DAMA*, *REI**
30 FOR X=1 TO 5: READ S:S$(X)=CH
R$(143+S): NEXT X
40 FOR X=1 TO 13:READ N$: N$(X)=
N$: NEXT X
45 CLS(6)
46 PT=0: CT=0
47 FOR X=1 TO  5: P(X)=0: C(X)=0
: NEXT
50 FOR X=1 TO 52: D(X)=X: NEXT
60 FOR X=1 TO  5: GOSUB 1000: P(
X)=Z: NEXT
70 FOR X=1 TO  3: GOSUB 1000: C(
X)=Z: NEXT
72 REM
75 REM IMPRESSAO DA 'MAO' DO JOG
ADOR
80 L=257
90 FOR M=1 TO 2: C=P(M): GOSUB 5
00: PT=PT+T: NEXT
100 FOR M=1 TO 3: S=5: GOSUB 200
0: NEXT
102 REM
105 REM IMPRESSAO DA 'MAO' DO CO
MPUTADOR
110 L=10
120 S=5: GOSUB 2000
130 C=C(2): GOSUB 500: CT=CT+T
150 PRINT @ 6, "'MAO' DO COMPUTA
DOR"
160 PRINT @ 267, "SUA 'MAO'"
200 L=269: K=3
205 PRINT @ 231, "OUTRA CARTA(S/
N)?";
210 R$=INKEY$: IF R$="" THEN 210
220 IF R$="N" THEN 255
230 C=P(K): GOSUB 500
240 PT=PT+T
242 FOR X=1 TO K
244 IF PT>21 AND (P(X)-1)/13=INT
((P(X)-1)/13) THEN PT=PT-10
246 NEXT X
247 IF PT>21 THEN PRINT @ 488, "
VOCE ESTOUROU!!!";: GOTO 400
250 K=K+1: IF K<6 THEN 205
255 L=10
260 C=C(1): GOSUB 500: CT=CT+T
360 IF PT <= CT THEN 380
370 PRINT @ 484,"PARABENS, VENCE
DOR!";
375 GOTO 390
380 PRINT @ 487, "VOCE DEU AZAR!
"
390 REM
400 PRINT @ 231, "OUTRO JOGO (S/
N)?";
410 R$=INKEY$: IF R$="" THEN 410
420 IF R$="S" THEN 45 ELSE END
430 IF N=1 THEN T=11
500 GOSUB 4000: GOSUB 2000
510 GOSUB 3000: RETURN
900 REM
1000 REM DANDO AS CARTAS
1005 Z=RND(52)
1010 IF D(Z)=0 THEN 1000
1020 D(Z)=0
1030 RETURN
1900 REM
2000 REM IMPRESSAO DOS NAIPES
2005 L1=L
2010 FOR X=1 TO 6
2015 L1=L1+32
2020 FOR Y=1 TO 5
2030 PRINT @ L1+(Y-1), S$(S);
2040 NEXT Y,X
2045 L1=0: L=L+6
2050 RETURN
2900 REM
3000 REM IMPRESSAO DOS VALORES D
AS CARTAS
3005 L1=L-6
3010 FOR X=1 TO 6
3020 L1=L1+32
3030 PRINT @ L1+2,MID$(N$(N),X,1
);
3040 NEXT X
3045 L1=0
3050 RETURN
3900 REM
4000 REM CALCULO DO VALOR E DO N
AIPE
4005 S=INT((C-1)/13)+1
4010 N=C-(S*13-13)
4015 REM CALCULO DOS PONTOS
4020 IF N=11 OR N=12 OR N=13 THE
N T=10 ELSE T=N
4030 IF N=1 THEN T=11
4040 RETURN
```

APÊNDICE K

Caleidoscópio

```
10 CLS0
20 X=RND(32)-1
30 Y=RND(16)-1
40 Z=RND(9)-1
50 GOSUB90
60 GOTO 20
90 IF Z=0 OR RND(7)=3 THEN 150
100 SET(31-X,16+Y,Z)
110 SET(31-X,15-Y,Z)
120 SET(32+X,16+Y,Z)
130 SET(32+X,15-Y,Z)
140 RETURN
150 RESET(31-X,16+Y)
160 RESET(31-X,15-Y)
170 RESET(32+X,16+Y)
180 RESET(32+X,15-Y)
190 RETURN
```

Dado eletrônico

```
4 CLEAR 2000
5 CLS(3)
6 DIM DB$(6)
8 DIM DF(21), P(6), D$(6)
10 REM
20 FOR X=1 TO 21
30 READ DF(X)
40 NEXT X
50 DATA 39
60 DATA 14, 64
70 DATA 14, 39, 64
80 DATA 14, 20, 58, 64
90 DATA 14, 20, 39, 58, 64
95 DATA 14, 20, 36, 42, 58, 64
100 REM
105 FOR X=1 TO 7
110 REM
120 REM POSICAO NO VETOR DF
130 FOR X=1 TO 6
140 READ P(X)
150 NEXT X
160 DATA 1, 2, 4, 7, 11, 16
165 REM
170 REM CONSTRUCAO DO STRING (DA
DO)
175 FOR X=1 TO 6
180 M=P(X)
185 FOR Y=1 TO 7
190 FOR Z=1 TO 11
192 IF (Y-1)*11+Z <> DF(M) THEN
200
194 D$(X)=D$(X) + CHR$(128)
196 M=M+1
197 IF M=22 THEN M=0
198 IF M=X THEN M=0
199 GOTO 230
200 D$(X)=D$(X)+CHR$(143+96)
230 NEXT Z
240 FOR Z=0 TO 31-11
250 D$(X)=D$(X)+CHR$(143+32)
260 NEXT Z
270 NEXT Y,X
480 REM
490 REM JOGANDO OS DADOS
500 FOR T=1 TO 10
510 A=RND(6): B=RND(6)
520 PRINT @ 35, D$(A);
530 PRINT @273, D$(B);
540 NEXT T
550 PRINT @ 80, "PRESSIONE  QUAL
-";
555 PRINT @112, "QUER  TECLA PAR
A";
560 PRINT @144, "A PROXIMA JOGAD
A";
570 K$=INKEY$: IF K$="" THEN 570
580 GOTO 500
```

APÊNDICE K

Gráficos de barra

```
10 DIM A(5,3,2), A$(5)
20 DATA UTILIDADES, PESSOAL, FOR
NECEDORES, ALUGEL, VIAGENS
30 FOR X=1 TO 5
40 READ A$(X)
50 CLS
60 PRINT @ 139, "DESPESAS"
70 PRINT @ 175 - INT(LEN(A$(X))/
2), A$(X)
80 PRINT
90 FOR Y=1 TO 3
100 PRINT "DEPTO." Y
110 INPUT "PREVISTAS"; A(X,Y,1)
120 INPUT "REAIS"; A(X,Y,2)
130 NEXT Y
140 NEXT X
150 CLS
160 PRINT @ 130, "O QUE VOCE GOS
TARIA DE VER"
170 L=203
180 FOR X=1 TO 5
190 PRINT @L, X; A$(X)
200 L=L+32
210 NEXT X
220 PRINT @ 460, "(1-5)"
230 INPUT X
235 C(1)=0: C(2)=0: LC(1)=0: LC(
2)=0
240 FOR Y=1 TO 3
250 C(1) = A(X,Y,1) + C(1)
260 C(2) = A(X,Y,2) + C(2)
270 NEXT Y
280 IF C(2) > C(1) THEN 310
290 LC(1)=30: LC(2)=INT(C(2)/C(1
)*30)
300 GOTO 320
310 LC(2)=30: LC(1)=INT(C(1)/C(2
)*30)
320 P=129
330 CLS(0)
340 PRINT @ 11, "DESPESAS";
350 PRINT @ 47 - INT(LEN(A$(X))/
2), A$(X);
360 PRINT @ 161, "PREVISTAS";
370 PRINT @ 321, "REAIS";
380 PRINT @ 448, CHR$(159)+CHR$(
159);
390 PRINT @ 451, "DEPTO.1";
400 PRINT @ 459, CHR$(175)+CHR$(
175);
410 PRINT @ 462, "DEPTO.2";
420 PRINT @ 470, CHR$(191)+CHR$(
191);
430 PRINT @ 473, "DEPTO.3";
440 PRINT @ 480, "PRESSIONE QQ T
ECLA P/CONTINUAR"
450 FOR M=1 TO 2
460 FOR N=1 TO 2
470 P1=P+32
480 FOR Y=1 TO 3
490 D(Y) = INT(A(X,Y,M)/C(1)*LC(
1))
500 FOR O=1 TO D(Y)
510 PRINT @ P1, CHR$(143+16*Y);
520 P1=P1+1
530 NEXT O
540 NEXT Y
550 P=P+32
560 NEXT N
570 P=289
580 NEXT M
590 K$=INKEY$: IF K$="" THEN 590
600 GOTO 150
```

APÊNDICE K

Tocando tons

```
10 DIM M(50), T(8)
20 FOR B=1 TO 8
30 READ T(B)
40 NEXT B
50 X=1
60 M(X)=RND(8)
70 FOR Y=1 TO X
80 CLS (M(Y))
90 PRINT @ 239, M(Y);
100 SOUND T(M(Y)), 8
110 NEXT Y
120 CLS
130 PRINT @ 234, "REPITA A NOTA"
140 FOR Y=1 TO X
150 T=1
160 K$=INKEY$
170 T= T+1
180 IF T>150 THEN 310
190 IF K$="" THEN 160
200 K= VAL(K$)
210 IF K<> M(Y) THEN 310
220 CLS(K)
230 PRINT @ 239, K;
240 SOUND T(K), 3
250 NEXT Y
260 X=X+1
270 CLS: PRINT @ 230, "OUCA A PR
OXIMA NOTA"
280 FOR T=1 TO 500: NEXT T
290 CLS: PRINT @ 230, "OUCA A PR
OXIMA NOTA"
300 GOTO 60
310 CLS(0)
320 PRINT @ 235, "VOCE PERDEU";
330 SOUND 1, 25
340 DATA  89, 108, 125, 133, 147
, 159, 170, 176
```

Compositor de música

```
5 DIM A(25), S$(13), B(200): Y=1
9 FOR X=1TO25:READA(X):NEXTX
20 DATA  89,  99, 100, 117, 125
30 DATA 133, 140, 147, 153, 159
40 DATA 165, 170, 176, 180, 185
50 DATA 189, 193, 197, 200, 204
60 DATA 207, 210, 213, 216, 218
70 FOR X=1TO13:READS$(X):NEXTX
80 DATA A,W,S,E,D,F,T,G,Y,H,U,J,
K
90 CLS: PRINT @ 165, "COMPONHA A
 SUA MUSICA"
94 PRINT @ 224, "USE AS TECLAS D
AS  2' E 3' FILAS"
96 PRINT @ 289, "PRESSIONE <X>
QUANDO TERMINAR"
100 P$=INKEY$
110 IF P$="" THEN 100
115 FOR X=1 TO 13
120 IF P$ <> S$(X) THEN 150
130 SOUND A(X), 5
140 B(Y)=X: Y=Y+1
150 NEXT X
160 IF P$ <> "X" THEN 100
165 CLS
170 PRINT @ 198, "REPRODUCAO DA
MUSICA"
174 PRINT @ 262, "QUE TECLA (1-1
1) ";
176 INPUT K
180 FORX=1TOY-1:SOUNDA(B(X)+K),5
190 NEXT X : GOTO 165
```

NOTAS