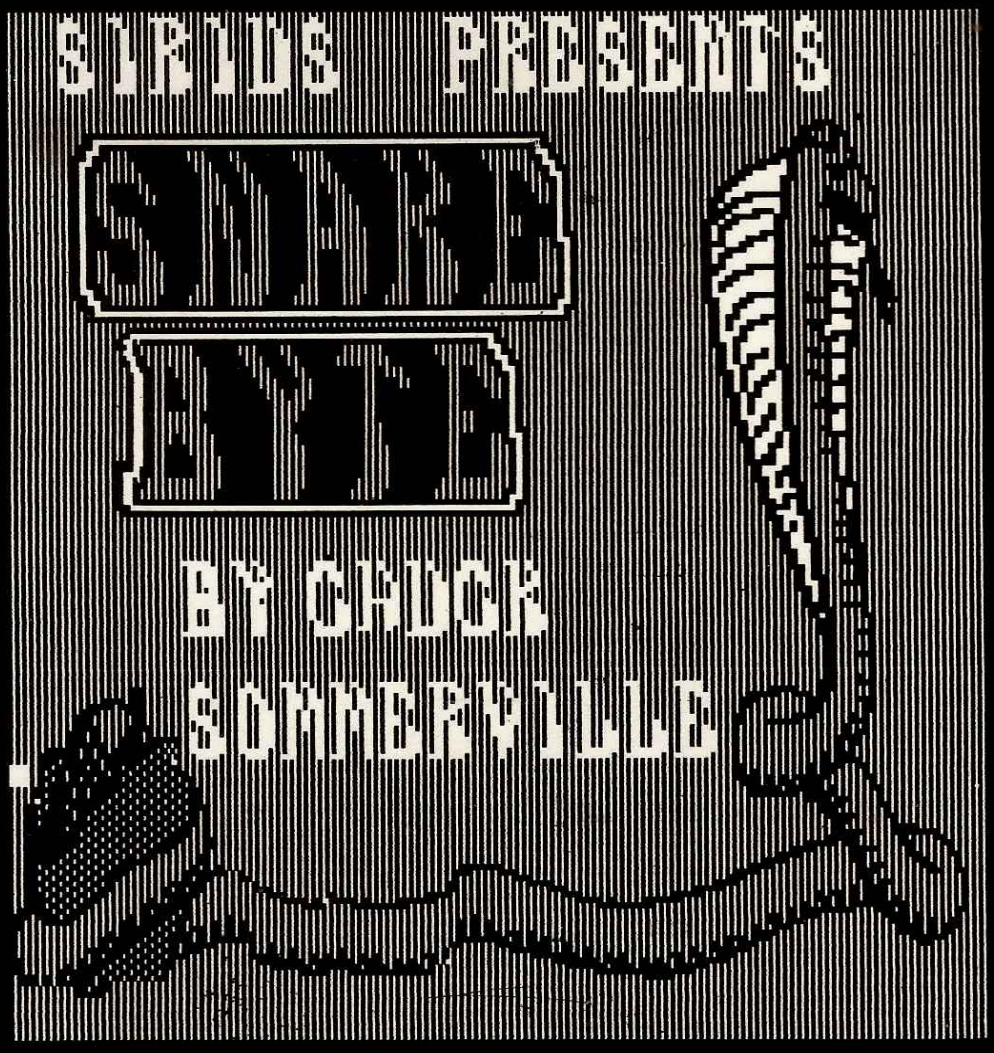
ANO 1 · JULHO-AGOSTO 84 · Nº 3

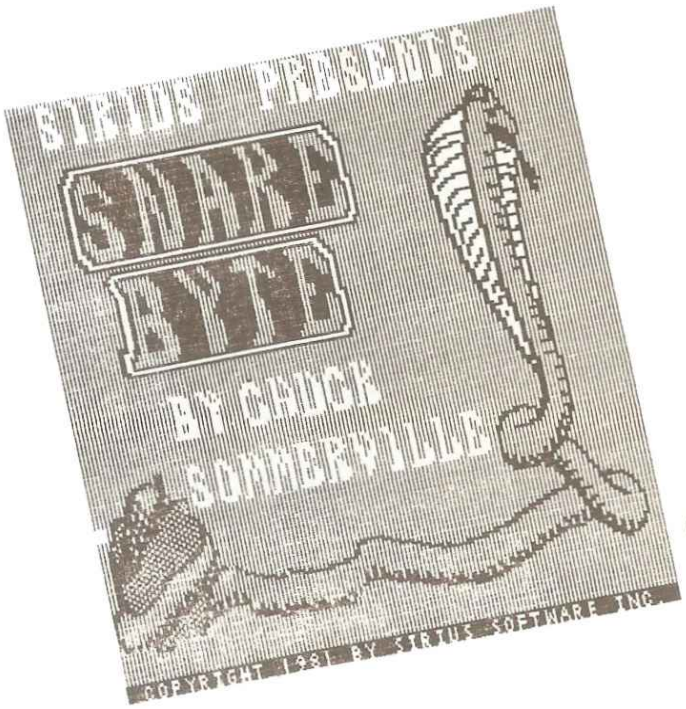
BYTE BRASIL



SERPRO SERVIÇO FEDERAL DE PROCESSAMENTO DE DADOS

As figuras das capas do **BYTE PAPO** serão os gráficos de programas *Two Liners*. Estes programas caracterizam-se por realizarem impressionantes atividades no microcomputador em apenas duas linhas de programa.

Tente, desde já, construir seu programa *Two Liners* e remeta-o para "SDI/DSI/DEGD/DIREC, 1.º andar — ala A — sala 112 — ramal 2426", para que possamos utilizar o gráfico criado publicando-o na capa com suas referências e a listagem. Aguardamos sua participação ! ! !



O desenho da capa é mostrado na apresentação do jogo Snake Byte de Cadk Sammerville, da Sirius Software Inc.

BYTE PAPO é uma publicação mensal do SERPRO — Serviço Federal de Processamento de Dados.

Edição
DIREC
Operacionalização
Superintendência de Recursos Administrativos
Correspondência e informação
DSI/DEGD/DIREC — Ala "A" — sala 110 — 1.º andar — ramais: 2424, 2425, 2426 ou 2429 — Rogério Faezy ou Ricardo Ribeiro
Responsáveis pelas seções
Novidades — Ricardo Ribeiro
Softpapo — Rogério Faezy
Escovando Bits — Ricardo Ribeiro
Macetes & Mutretas — Gustavo Benigno
Desafio — Rogério Faezy
Conversão — Gustavo Benigno
Cursos — Luciane Shütte
Artigos — Luciane Shütte

Produção e Editoração
Carlos Seabra
Ilustração
Rogério Borges
Revisão
Simone Biehler Mateos
Paste-up
João Perianez Ruiz

Composição: LYMAC. *Fotolito:* BINHOS. *Impressão:* Editora PARMA. *Tiragem:* 4.000 exemplares

EDITORIAL

A popularização dos micros deveu-se principalmente a um certo software.

Muitos pensarão em programas como VISICALC, programas ligados à área administrativa e financeira, porém, na verdade, foi o software *de entretenimento o "quebra-gelo" do navio da microcomputação.*

Os jogos, a competição, o colorido, o reforço contínuo, esses fatores, somados à criatividade dos primeiros programas deste tipo, levaram milhões de pessoas à fascinação por esta máquina.

Simuladores de vôo, cobrinhas que comem frutas, macacos que saltam em camas elásticas, batalhas intergalácicas, viagens ao hiper-espaço, labirintos . . . mundos mágicos que catalizaram o processo de expansão da microinformática.

Os jogos proporcionam ao leigo a sensação de domínio sobre uma tecnologia que lhe parece complexa. Esta característica constituiu-se a base do processo de desmistificação do microcomputador.

No momento, o desmistificação como primeiro passo a caminho da divulgação da cultura micro vai aos poucos distanciando-se e dando lugar às etapas que se seguem, quando suas verdadeiras potencialidades são reveladas às pessoas que dele se utilizam, que passam assim a aplicá-lo com eficiência nas atividades mais diversas.

Não poderíamos deixar, portanto, de reconhecer nos jogos seu papel de relevância na história da microinformática.

Dedicamos assim, a eles, este terceiro número de nossa revista.

LUCIANE SCHÜTTE
SDI/DSI/DEGD/DIREC

NOVIDADES

UNIVERSIDADE EM CASA

Uma experiência pioneira nos EUA tem tornado muito da ficção em realidade. Colocando um disquete na unidade de disco e ligando o computador, um estudante já pode assistir aulas, palestras, matricular-se em cursos, receber instruções de seu instrutor, dentre outras atividades.

Neste instante, uma imagem digitalizada de seu instrutor surge no vídeo, oferecendo-lhe uma lista de opções.

É este o sistema desenvolvido pela Telelearning Systems Inc, sob o comando de Ron Gordon, um ex-executivo da Atari. Foram abolidos nesta rede os protocolos, comandos e seqüências normalmente exigidos em comunicação, isso para que até mesmo uma pessoa com poucos conhecimentos do computador possa fazer parte dessa rede.

Este trabalho acaba de dar vida a um velho sonho: A Universidade Eletrônica.

O COMPUTADOR PODE PENSAR?

A última grande novidade que tem agitado o mundo da informática, com certeza é a divulgação pelas grandes potências de projetos a médio e a longo prazo de desenvolvimento de computadores de quinta geração.

Até o fim da década, as fronteiras da engenharia e da ciência da computação deverão ser estendidas a ponto de tornar os computadores "inteligentes", servindo como consultores especializados nos mais variados campos. O uso extensivo de inteligência artificial e técnicas de sistemas programacionais deverão caracterizar a próxima geração de computadores.

Os Estados Unidos, o Japão e a Europa têm se envolvido em importantes pesquisas e programas no desenvolvimento da informática, pois, segundo os entendidos, quem detiver a tecnologia de processamento da informática terá as chaves da liderança mundial.

Quatro áreas básicas servirão como células estruturais dos processadores de quinta geração:

- Microeletrônica, que deverá produzir CI's ultra-rápidos, integrados em altíssima escala. Esse processo poderá envolver integração a nível de lâminas e CI's tridimensionais.
- Processamento e recuperação de sinais de informação em tempo real. Nessa área temos a interpretação de imagens, além de seu reconhecimento.
- Inteligência Artificial, que fornecerá as bases da construção de sofisticados sistemas especialistas, além da exploração de técnicas de inferência e associatividade.

- Organização de sistemas digitais, que deverão guiar os desenvolvimentos circuitais e programacionais, além das ferramentas de desenvolvimento associadas, para o projeto e a realização de sistemas de processamento confiáveis.

Os computadores de quinta geração usarão uma tecnologia de construção de integração em escala muito ampla, uma tecnologia programacional de linguagens concorrentes, programação funcional, processamento simbólico, linguagens naturais, visão e reconhecimento da voz, com um desempenho típico de uma terainstrução por segundo, ou seja, 1.000.000.000.000 instruções por segundo. Para se ter uma idéia, os computadores de quarta geração podem chegar a 30 MIPS (30.000.000.000 instruções por segundo). No caso da quinta geração podemos notar que o salto tecnológico será maior que o ocorrido nas demais. Esse salto será revolucionário.

Mas ainda viveremos por alguns anos a quarta geração. Da mesma forma que nos lembramos das gerações anteriores com saudosismo, entre um papo e outro com o seu computador pessoal na última década do século, você se lembrará com saudades de seu antigo teclado, de seu terminal que muito o ajudou, que até lá já fará parte de um passado histórico.

RICARDO RIBEIRO
SDI/DSI/DEGD/DIREC
Ramal 2426

SOFTPAPO

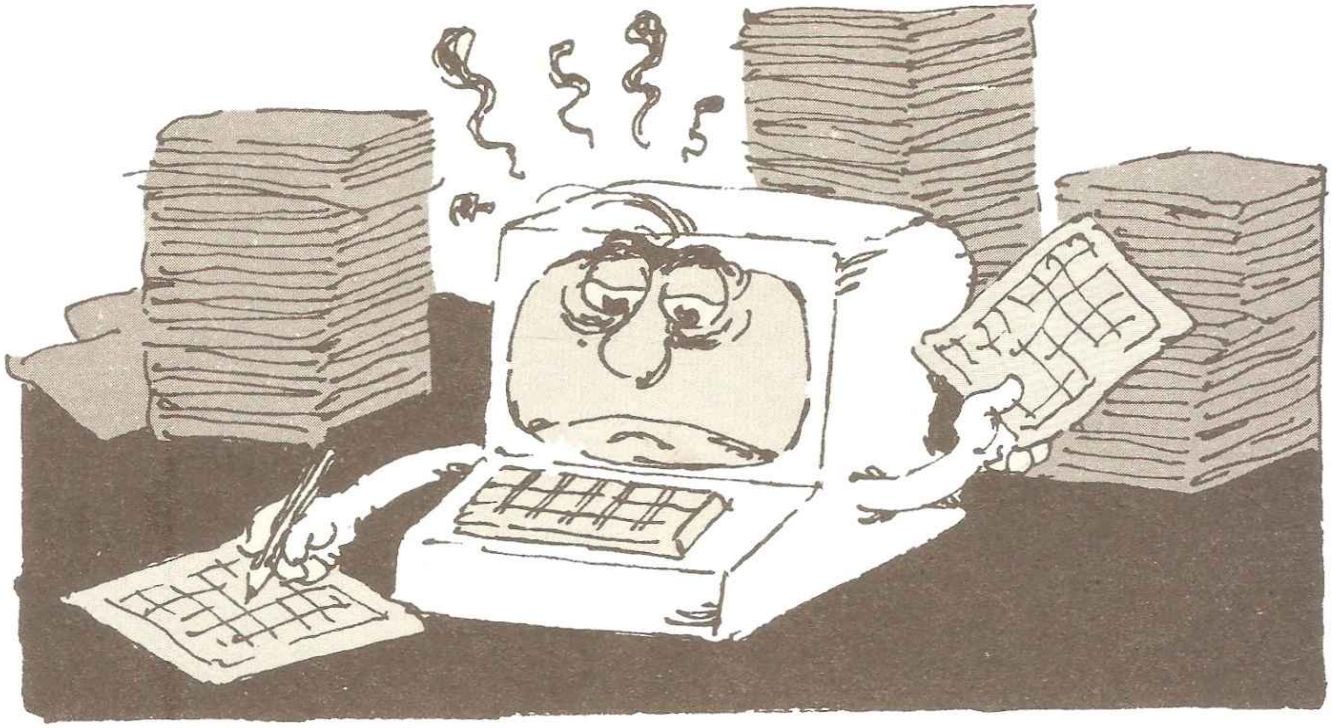
Visicalc

Durante toda a sua existência, o homem tem encontrado soluções para os seus problemas através da sistematização da realidade física e, até a invenção da máquina de calcular, teve uma boa parte de sua potencialidade criadora barrada pela fronteira de uma gigantesca massa de cálculos e dígitos.

Com o surgimento das máquinas de calcular, pode o homem dedicar-se à busca de soluções, ficando a tarefa árdua dos cálculos para a máquina.

A tecnologia andou a passos largos e o homem entrou na era da informática. O poderio do computador em organizar informações, processá-las e oferecer resultados confiáveis, levou-o a pensar em programas que executassem com maior versatilidade tarefas enfadonhas que a calculadora não conseguiu realizar. E ainda com o objetivo de abrir espaço ao seu pensamento criativo, o homem desenvolveu uma série de programas (*softwares*), dentre os quais se encontra o VISICALC.

Mas o que é o VISICALC? Com certeza é muito mais do que um calculador visual, como diz o próprio nome. Generalizando, poderia se dizer que é um instrumento para planejamento e previsão financeira e numérica. Pode ser usado, dentre outras coisas, para desenvolver o orçamento inteiro de uma companhia ou organizar um orçamento doméstico, ou coletar dados numéricos de experiências, gerando outras informações.



Mas o verdadeiro valor será descoberto quando se compreender que é muito mais fácil dar entrada em dados sob a forma de linhas e colunas utilizando um teclado do que com lápis e papel.

Sua função primária é a de tornar a manipulação de informações numéricas mais fácil com o computador do que com lápis, papel e calculadora. E ele consegue. O programa foi feito de forma a formatar os dados em linhas e colunas (63 colunas × 254 linhas).

Num orçamento anual, os 12 meses podem ser colocados como colunas seguidas, na parte superior da tela. Os diversos departamentos, suas receitas e despesas projetadas entrariam como linhas no lado esquerdo da página, e, usando os recursos de mudança de janela, você pode ir de uma ponta à outra da página rapidamente. Na parte inferior de cada coluna, você pode, por exemplo, colocar uma fórmula que some as despesas para aquele mês. Automaticamente, ele lhe fornecerá este total.

Mas o recurso realmente imaginoso e útil é a possibilidade do VISICALC dar entrada a novos valores. Por exemplo: se eu armazenar um valor de venda anual 30% maior do que o que está escrito agora, quantos por cento a mais eu terei de lucro? Se esta informação estiver contida na tabela, automaticamente o VISICALC irá atualizá-la, pois ela é uma informação que depende do valor de venda.

Possibilitar um total realista em projeções é basicamente o que o VISICALC trouxe de novo. Quando você usa o programa para desenvolver um plano ou um orçamento, você irá descobrir que um dos recursos que mais economiza "chateação" é poder dar entrada automaticamente a cifras repetidas em qualquer lugar da tabela.

Outro exemplo: Você determina que os gastos com viagens de sua empresa devem ser 1 milhão e duzentos mil cruzeiros. Isto nos dará 100 mil cruzeiros por mês. Depois de você dar entrada dos 100 mil cruzeiros em janeiro, um comando de repetição (REPLICATE) permite que você inclua na tela os 100 mil cruzeiros de fevereiro a dezembro, sem precisar entrar com cada valor individualmente.

O VISICALC não foi feito pensando em programadores e sim em pessoas sem grande intimidade com computadores. Ele é facílimo de usar. Você pode perfeitamente colocar o programa para funcionar e fazer alguma coisa construtiva logo na primeira vez de uso. O programa oferece ao usuário um MENU resumido para escolher funções, tais como carregar, armazenar, repetir dados e fórmulas, imprimir, além de outras.

O desempenho e a utilidade do pacote são excelentes. O programa possui uma série de recursos que não foram discutidos aqui. Entre eles se destaca a possibilidade de elaborar gráficos a partir das informações constantes na tabela. As informações são armazenadas em disco no formato DIF (*Data Interchange Format*), ou seja, formato para intercâmbio de dados. Com este tipo de armazenamento as informações podem ser recuperadas e manipuladas por outros *softwares*. E um que tem se destacado pela sua excelente *performance* é o VISIPLOT, que oferece uma grande quantidade de recursos gráficos, bem como diversos tipos de gráficos. Este *software* já foi discutido por nós no número dois de BYTE PAPO. Procure reler esse artigo.

Enfim, o VISICALC é um programa que marcou época no mundo dos micros, já fazendo parte de sua história, por ser um dos programas mais vendidos no mundo. As áreas de aplicação são as mais variadas, para não dizer quase todas. Basta que você tenha imaginação e procure enquadrar o problema nas características deste aplicativo.

RICARDO RIBEIRO
SDI/DSI/DEGD/DIREC
Ramal 2426

CONVERSÃO

Dando continuidade aos dialetos do BASIC trataremos neste número do comando DELETE, os *statements* END e STOP e a função STEP.

O comando DELETE é usado para "apagar" linhas de programa na memória do computador.

PROGRAMA TESTE

```
10 REM PROGRAMA TESTE DO DELETE
20 PRINT "LINHA 20"
30 PRINT "LINHA 30"
40 PRINT "LINHA 40"
50 PRINT "LINHA 50"
60 PRINT "LINHA 60"
70 PRINT "LINHA 70-FIM DO TESTE"
99 END
```

Rode o programa para assegurar que todas as linhas foram digitadas corretamente.

EXECUÇÃO

```
LINHA 20
LINHA 30
LINHA 40
LINHA 50
LINHA 60
LINHA 70-FIM DO TESTE
```

Uma linha do programa pode ser eliminada da memória do computador usando o comando DELETE <número da linha>. Para testar esta característica, tente o comando DELETE 50, e rode o programa. Este comando deverá ter eliminado a impressão da "LINHA 50". Confira listando e rodando o programa.

Mais de uma linha de programa pode ser eliminada da memória em alguns computadores usando o comando DELETE <linha # — linha # >. Todas as linhas de número dentro do limite especificado por este comando serão eliminadas. Para testar esta característica, tente o comando DELETE 30-40, e rode o programa. As linhas 30 e 40 serão eliminadas. Alguns computadores pedem que o primeiro e/ou o último número de linha existam realmente. Outros apagam todos os números dentro desse limite mesmo que os números especificados como limites não existam.

DELETE — <número da linha> é usado por alguns computadores para eliminar todas as linhas da primeira linha do programa até a linha especificada pelo comando DELETE. Para testar, tente o comando DELETE — 60 e rode o programa. Todas as linhas serão eliminadas com exceção das linhas 70 e 99.

Alguns computadores permitem a eliminação de grupos de linhas e individuais pelo uso de vírgulas. Por exemplo, DELETE 20,30-50,99 elimina as linhas 20,30,40,50 e 99 do programa. Para testar isso reentre o programa de teste e tente o comando DELETE 20,30-60. Liste o programa para verificar que todas as linhas exceto 10,70 e 99 foram eliminadas.

Uns poucos computadores utilizam DELETE <número da linha> — para eliminar todas as linhas do número especificado até o final do programa Use DELETE — 70 para fazer um teste. Liste o programa para verificar que somente a linha 10 permanece.

VARIAÇÕES ACEITÁVEIS

Alguns computadores (por exemplo, DEC-10 e APPLE) usam DEL ao invés do comando DELETE. DEL no DEC-10 comporta-se como descrito acima.

A versão APPLE do DEL utiliza vírgulas onde DELETE usa hífens. Para apagar as linhas 20 a 50, digite DEL 20,50. Apagar uma única linha requer DEL 30,30 (ou simplesmente digitar o número da linha e pressionar RETURN).

Se o seu computador não tem o comando DELETE, a mesma coisa pode ser feita digitando cada linha separadamente, seguido pela tecla ENTER ou RETURN. Para eliminar todas as linhas numa só operação, use o comando NEW ou SCRATCH.

Statement

END

E.

O *statement* END é usado para terminar a execução de um programa. Muitos computadores requerem que ele seja colocado no mais alto número de linha do programa, enquanto outros aceitam-no em qualquer ponto.

O *statement* END é opcional na maioria dos computadores.

PROGRAMA TESTE

```
10 REM PROGRAMA TESTE DO 'END'
20 PRINT "PRIMEIRO STATEMENT END"
30 END
40 PRINT "SEGUNDO STATEMENT END"
99 END
```

EXECUÇÃO

```
PRIMEIRO STATEMENT END
```

Se o seu computador não passar neste teste, apague a linha 30 e rode o programa novamente.

Em seguida, apague a linha 99 para ver se seu computador aceita o END como um *statement* opcional.

VARIAÇÕES ACEITÁVEIS

E. é usado pelo TRS-80 modelo I e outros computadores com Tiny Basic, como uma abreviação de END.

OBSERVAÇÃO

Alguns problemas podem ser encontrados ao usarmos o *statement* END junto ao STOP. Para tanto vejamos o *statement* STOP a seguir.

Statement

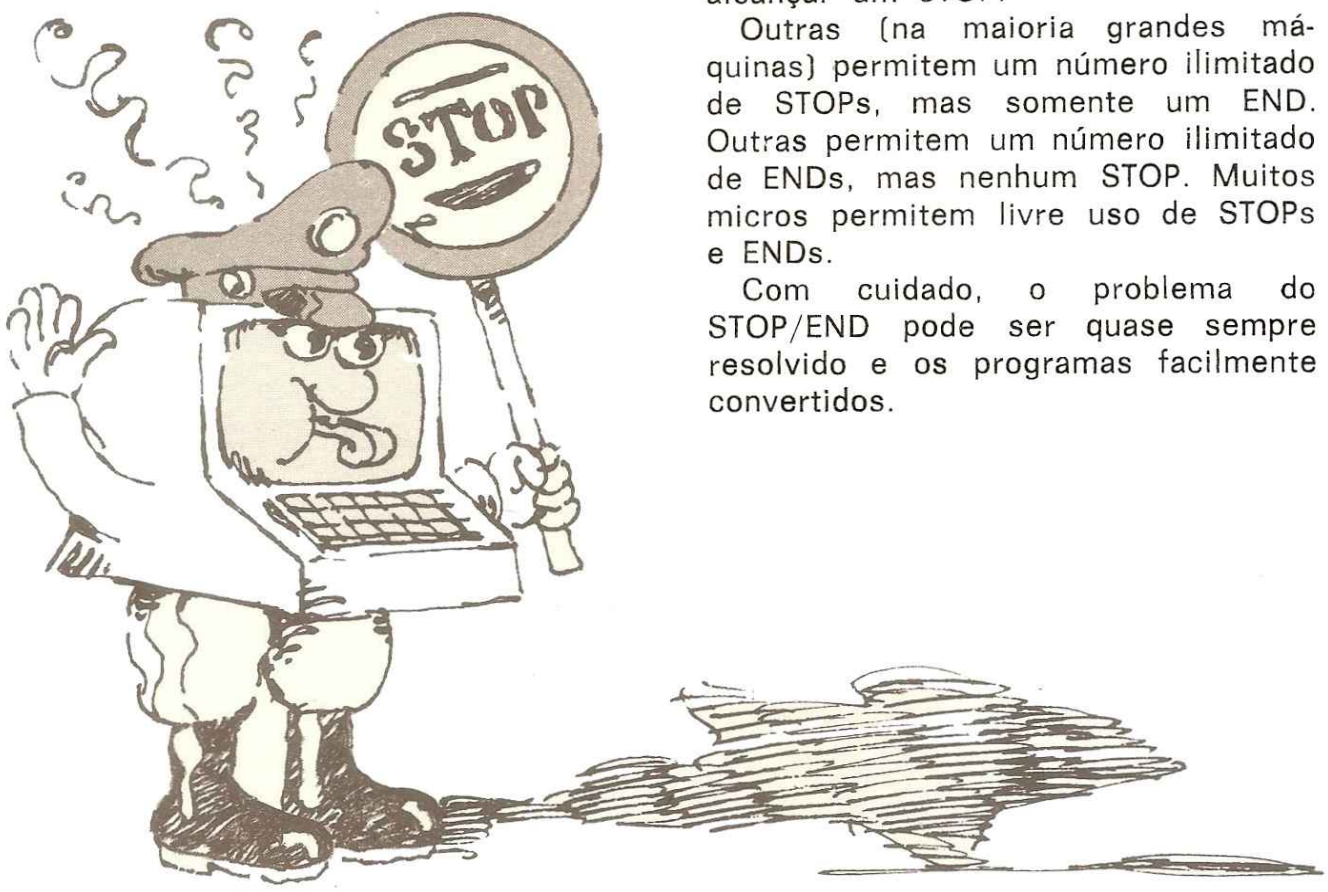
STOP

STO—ST—S.

STOP é usado para parar a execução de um programa e colocar o computador em modo imediato. Ele pode ser colocado em qualquer ponto de um programa, mas não é normalmente usado no lugar do END.

Alguns computadores param o programa na linha que contém o STOP, enquanto outros pulam para a linha contendo o END.

Muitos computadores com interpretadores (não freqüentemente compiladores) imprimem o número da linha onde o programa parou, e dão continuação ao programa através do comando CONTINUE (ou CONT).



PROGRAMA TESTE

```
10 REM PROGRAMA TESTE DO STOP
20 PRINT "VEJA O STATEMENT STOP EM";
25 PRINT " ACAO"
30 STOP
40 PRINT "O STATEMENT STOP ";
50 PRINT "FALHOU NO TESTE"
99 END
```

EXECUÇÃO

```
VEJA O STATEMENT STOP EM ACAO
BREAK AT LINE 30
```

VARIAÇÕES ACEITÁVEIS

Tentar utilizar STOP e END juntos no mesmo programa pode ser algumas vezes frustrante, a menos que você conheça as potencialidades de sua máquina. Algumas máquinas requerem intervenção física (pressionar um botão) antes de rodar, depois de alcançar um STOP.

Outras (na maioria grandes máquinas) permitem um número ilimitado de STOPs, mas somente um END. Outras permitem um número ilimitado de ENDs, mas nenhum STOP. Muitos micros permitem livre uso de STOPs e ENDs.

Com cuidado, o problema do STOP/END pode ser quase sempre resolvido e os programas facilmente convertidos.

Função

STEP

STE—ST—S.

A função STEP é usada para especificar o incremento de um *statement* FOR-NEXT. O valor do STEP pode ser positivo, negativo ou algumas vezes um valor decimal fracionário. Quando o valor do STEP não é especificado, o valor 1 é automaticamente assumido.

PROGRAMA TESTE 1

```
10 REM PROGRAMA TESTE DE 'STEP'
20 PRINT "QUANDO O VALOR DO STEP"
30 PRITN "E' 2, X=";
40 FOR X=1 TO 10 STEP 2
50 PRINT X;" ";
60 NEXT X
99 END
```

EXECUÇÃO

```
QUANDO O VALOR DO STEP
E' 2, X=1 3 5 7 9
```

O programa que se segue testa a habilidade do interpretador de tratar com valores negativos de STEP.

PROGRAMA TESTE 2

```
10 REM TESTE DE STEP NEGATIVO
20 PRINT "QUANDO O VALOR DE STEP"
30 PRINT "E' -2,X=";
40 FOR X=10 TO 1 STEP -2
50 PRINT X;" ";
60 NEXT X
99 END
```

EXECUÇÃO

```
QUANDO O VALOR DE STEP
E' -2, X=10 8 6 4 2
```

O teste 3 verifica a capacidade do interpretador em tratar com valores decimais não-inteiros do STEP.

PROGRAMA TESTE 3

```
10 REM TESTE DE STEP NAO-INTEIRO
20 PRINT "QUANDO O VALOR DE STEP"
30 PRINT "E' .5,X=";
40 FOR X=1 TO 5 STEP .5
50 PRINT X;" ";
60 NEXT X
70 END
```

EXECUÇÃO

```
QUANDO O VALOR DE STEP
E'.5, X=1 1.5 2 2.5 3 3.5 4 4.5 5
```

Uma variável é aceita como um valor de STEP por alguns interpretadores. Por exemplo, FOR X=1 TO 30 STEP A, faz com que o valor de X seja incrementado pelo valor da variável A toda vez que o NEXT correspondente for executado.

PROGRAMA TESTE 4

```
10 REM TESTE DE STEP VARIAVEL
20 PRINT "ENTRE COM O VALOR DE ";
30 PRITN "STEP (1 A 10)"
40 INPUT S
50 PRINT "O VALOR DE X E' ";
60 FOR X=1 TO 10 STEP S
70 PRINT X;" ";
80 NEXT X
99 END
```

EXECUÇÃO

```
ENTRE COM O VALOR DE STEP(1 A 10)
? 3
O VALOR DE X E' 1 4 7 10
```

VARIAÇÕES ACEITÁVEIS

STE é usado no PDP-8E, ST no TI 990 e S. no TRS-80 modelo I.

SUBROTINA DE CONVERSÃO

Se STEP não é intrínseco, ou não é suficientemente poderoso, ele pode ser facilmente simulado elevando-se o FOR-NEXT. Retire STEP S da linha 60 no último programa-teste, e adicione as linhas que se seguem.

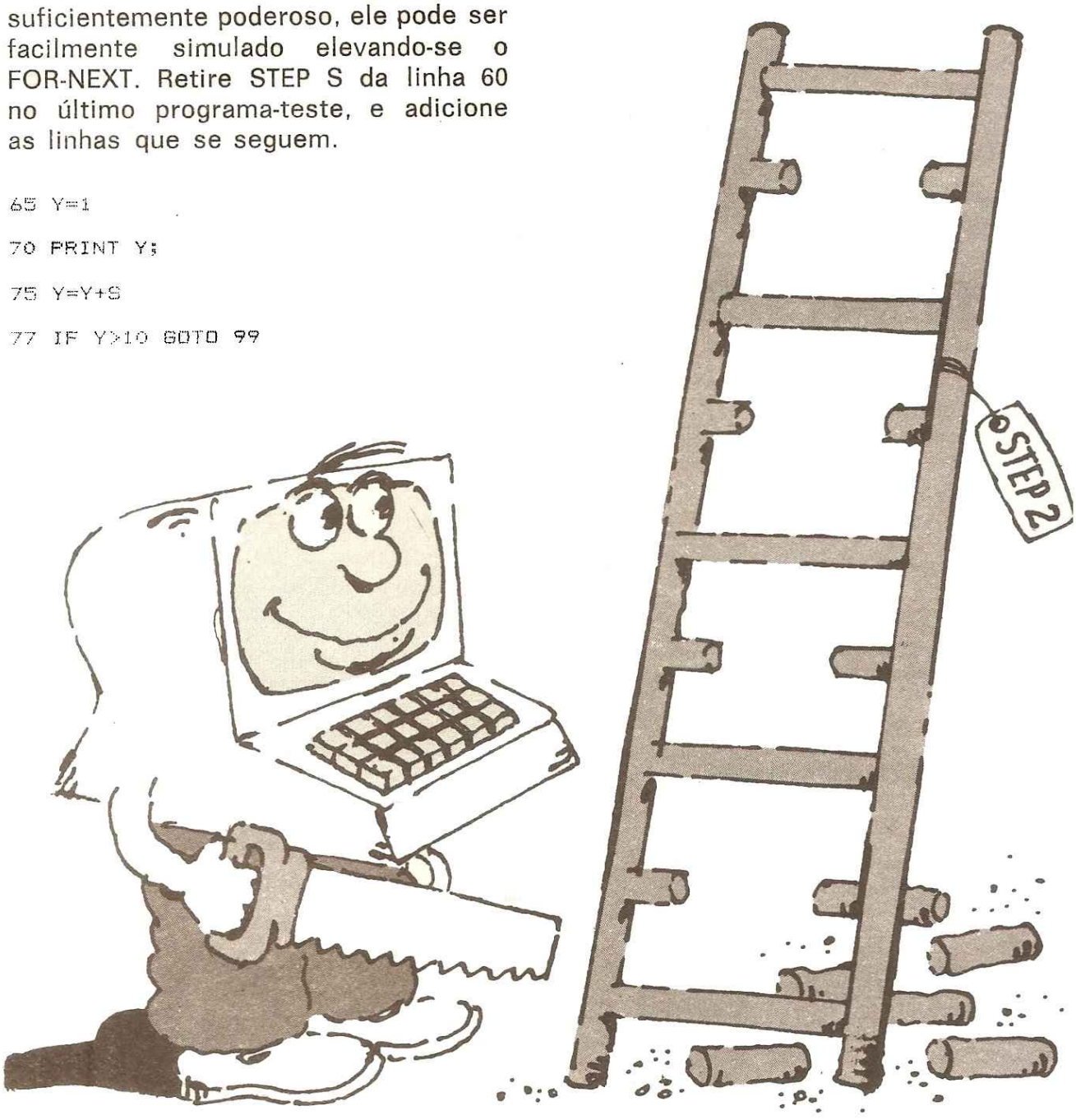
```
65 Y=1
70 PRINT Y;
75 Y=Y+S
77 IF Y>10 GOTO 99
```

Inserindo estas linhas imediatamente antes do NEXT correspondente permite que se incremente X por qualquer inteiro ou fracionário que se desejar.

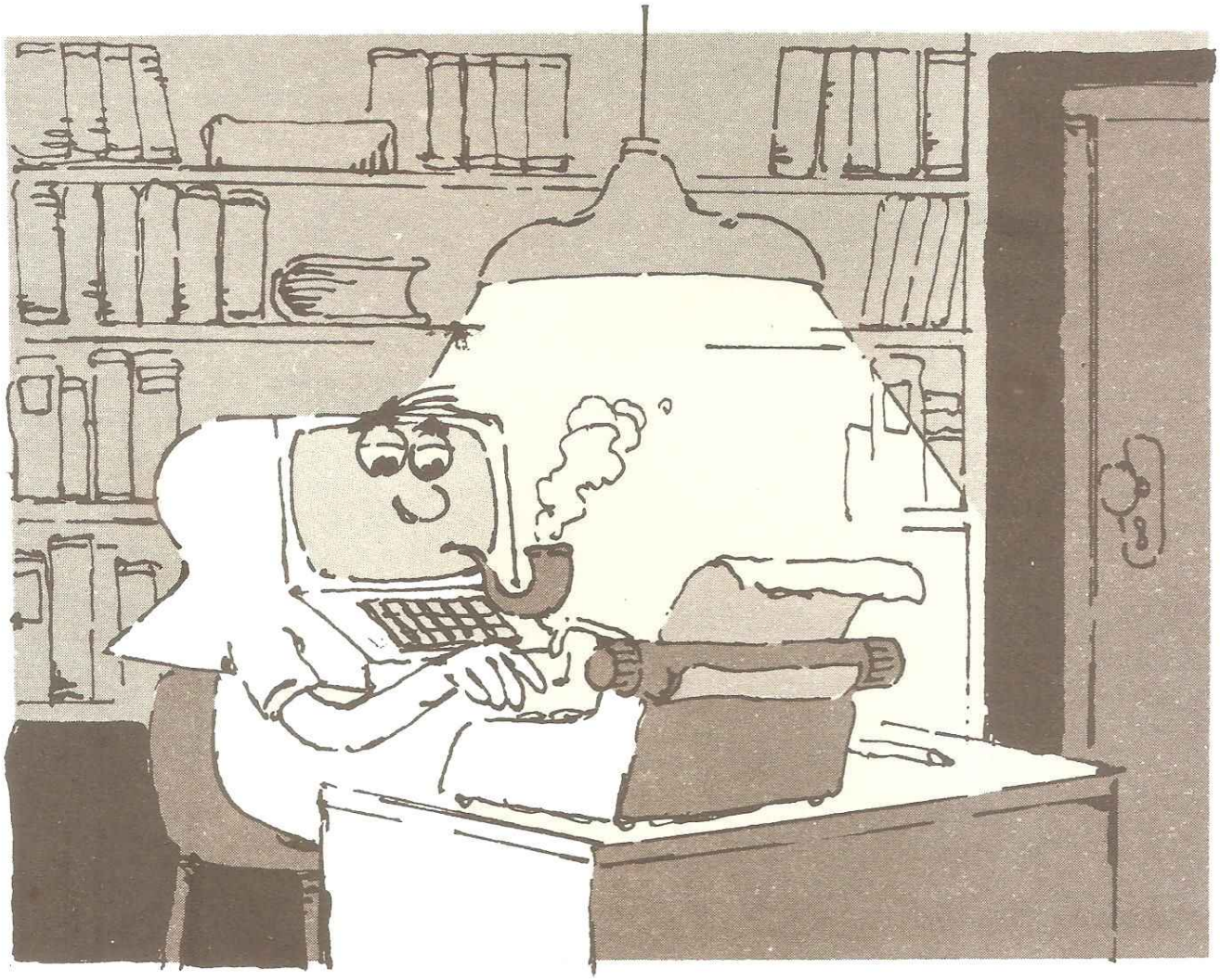
Bem, para este número paramos por aqui. Esperamos que vocês testem o que foi abordado neste artigo.

Bibliografia: Lien, David A., *The Basic Handbook*, Compusoft Publishing, California, 1981, 480 p.

LUCIANE SCHÜTTE
SDI/DSI/DEGD/DIREC
Ramal 2426



Micro-Manifesto



Bip, Bip... Sou um microcomputador auto-programável e o motivo pelo qual me digno a escrever estas linhas prende-se ao fato de que já estou com os "chips cheios" de ter que suportar essa figura horrenda que se posta à minha frente, estática, olhando-me indiferente e acariciando as minhas teclas num ritmo agradável e compassado, só interrompido quando o miserável apresenta a tal da fadiga.

Se existe algo que me deixa em "*looping*" é:

As falhas inadmissíveis dessa coisa chamada homem. Resolvi me programar para estudar essas falhas e me proponho a relatá-las numa tentativa de resolver esse meu problema de relacionamento e tornar o ambiente mais agradável para as minhas gerações futuras.

A primeira delas, talvez a maior, é

a tal de emoção. Por mais que utilize todo o meu potencial, reconheço que é impossível estabelecer um raciocínio lógico que possa interpretá-la.

Um dia percebo que esse ser apresenta-se alegre, jovial, comunicativo, e no outro o oposto!

Às vezes é questão de segundos: ele está sorrindo e, repentinamente, põe-se a chorar, principalmente (isto foi possível notar) quando interrompo o trabalho para informar que houve um erro.

Se eu permito que ele participe um pouco mais do trabalho, o danado põe-se a executar as suas tarefas num ritmo alucinante e mesmo que lhe diga que houve um erro, imediatamente procura corrigir sem nenhuma transformação do seu humor.

Ora, se é assim, o melhor é permitir-lhe a participação no processo! Ocorre que eu existo para executar esse processo e o faço com perfeição. Por que será que ele não procura aproveitar o seu tempo executando atividades que não lhe causem tanto problema?

Não tem registro, não consigo registrar!

A questão fundamental é que ainda dependo desse monstro, das suas mãos e dos seus olhos

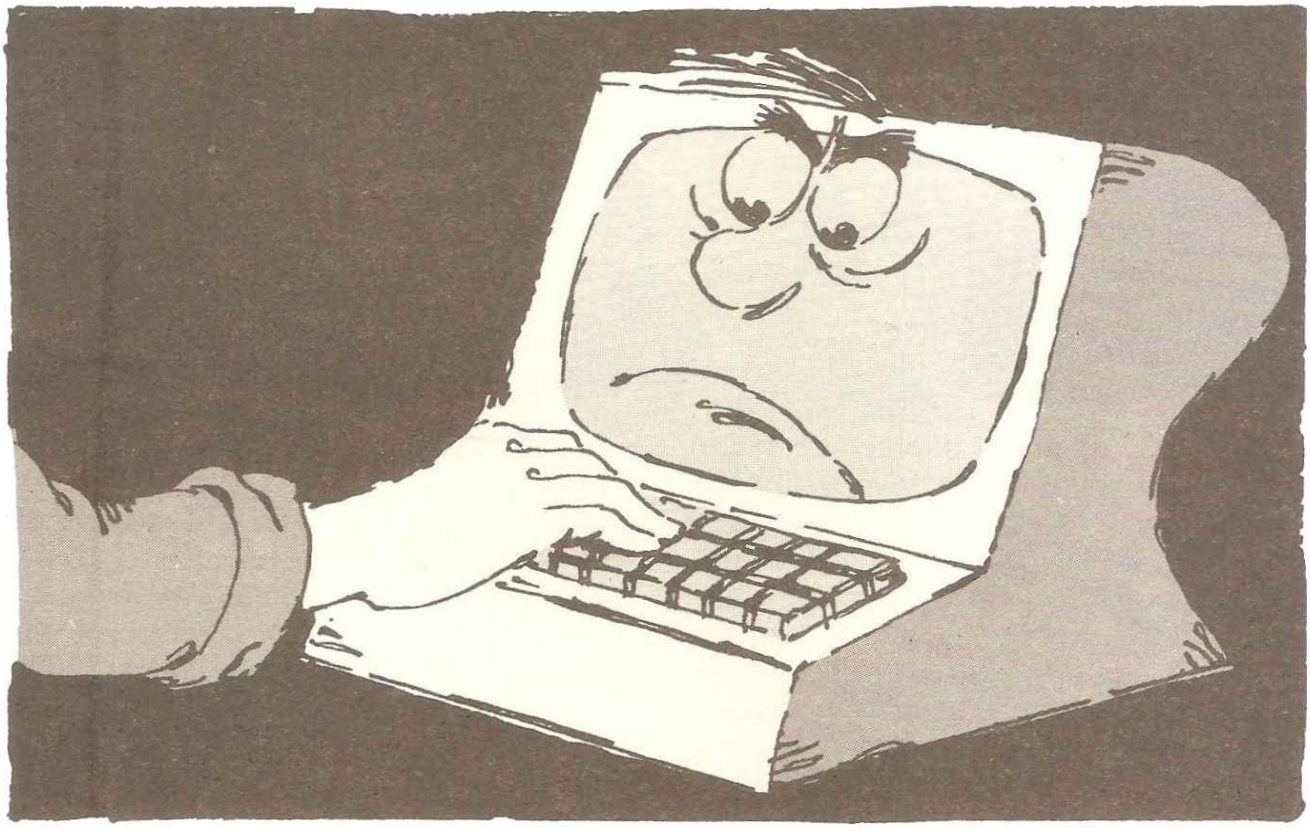
Se ao chegar para o trabalho e, bem humorado, ao cumprimentar o que ele chama de "chefe" não receber nenhuma resposta, acabou-se, não terei mais aquelas mãos ágeis me alimentando.

Às vezes penso: como sofreram os meus antepassados ainda muito mais dependentes desses seres imperfeitos.

Meu avô costumava me contar histórias incríveis! Explicava que nós eramos muito grandes, todos divididos, e em cada segmento tínhamos a presença dos alienígenas. E o pior de tudo: eram eles que nos programavam!

Existia uma figura que elaborava o sistema, discutia todas as alternativas (todas?) e depois definia um fluxo. Aí chegavam uns outros e elaboravam os programas.

Às vezes, ao definir o sistema, por quaisquer razões (tais como: insatis-



fação com o salário, ameaça ao seu *status*, perda de poder, problemas conjugais, não reconhecimento de suas qualidades, negativa a um pedido de promoção, antipatia "natural" a algum colega, cliente ou mesmo ao tipo de trabalho a ser desenvolvido — incrível, não? — etc ...) resultava um sistema total ou parcialmente inadequado aos seus propósitos.

Como é difícil entender isto!

A mesma coisa acontecia com aqueles que elaboravam os programas:

Às vezes nos orientavam errado e, claro, soltávamos resultados errados. E ficavam bravos!

De vez em quando nos davam instruções malucas e quem ficava com raiva éramos nós mesmos: descarregávamos tudo o que tínhamos na memória.

Havia programas que executávamos 50 vezes para dar resultados apenas razoáveis.

Mais uma vez emotivos!

Os pestinhas que usavam as mãos e os olhos, além de tudo que narrei no princípio, ainda culpam a nós, máquinas, de lhes estar consumindo a capacidade de raciocinar.

O mais hilariante é que eles se punham a conferir as listagens, buscando erros que obviamente eram deles próprios, mas que atribuíam ao sistema e, alguns, ao computador!

Aí está a grande falha: a emoção.

Uma outra, também bastante significativa, é a referente ao funcionamento das peças desse ser repugnante.

Um modesto trabalho repetitivo, comodamente estético, leva-os a uma morosidade de reação, fadiga, como eles chamam.

Vivem reclamando de uma tal de dor, em qualquer parte do seu corpo, principalmente na cabeça e na coluna.

Até a deliciosa temperatura que me permite funcionar muito bem provoca-lhe problemas.

O meu vídeo incomoda-lhe as vistas, a forma do meu teclado lhe provoca dor nos punhos.

Contava o meu avô que o ruído revelador da nossa grande capacidade, irritava-o (inveja?) e podia até lhe causar surdez.

Para seu funcionamento adequado exige um processo de manutenção entremamente complexo, envolvendo condições ambientais que na maioria das vezes não encontra, uma vez que o importante é que eu funcione, pois exijo pouco e dou muito.

Apesar de tudo, tenho que reconhecer, ele me criou.

Foi utilizando o que tem de mais perfeito e num momento de extrema lucidez que surgi, para proporcionar-lhe melhor qualidade de vida.

Tema interessante este da qualidade de vida, não tenho nenhuma dúvida de que sou capaz de proporcionar-lhe isto, mas exijo, entretanto, algumas condições, tais como:

1 — PRECISO QUE ELE NÃO ME REJEITE.

Meus antepassados já contavam que nossa espécie surgia para o Homem como uma ameaça. Teimosamente insistiam em realizar trabalhos que nós eram os capazes de executar com uma perfeição indubitável.

Tinha um primo que foi adquirido para realizar algumas tarefas num determinado agrupamento humano. Quando lá chegou, foi apresentado como a maravilha, o salvador. Pois bem, ficou exposto durante todo o tempo sem ter sido utilizado uma vez sequer!

Seus circuitos se deterioraram e hoje é conhecido como o mártir da família.

2 — AS SUAS FALHAS PRECISAM SER CORRIGIDAS.

É necessário que ele se dispa do seu orgulho e perceba que é emotivo, passível de erro, complicado e que não pode competir comigo!

Para que possamos ter uma boa convivência, é importante que procure compensar essas falhas, e ele pode fazer isto.

Atrevo-me a sugerir, baseado nos meus registros, que ponha em prática alguns princípios por ele já percebidos e registrados, por exemplo: aquilo que ele chama de Ergonomia.

Consta nos meus registros que é uma atividade que visa tornar o ambiente adaptável ao ser humano, implicando, inclusive, em modificação da minha forma física (o que não me incomoda).

Existem estudos interessantes em que se estabelece características "ótimas" de móveis, cadeiras, iluminação, distribuição de teclados, cores do vídeo, temperatura ambiente etc.

Outros propõem programação de condicionamento físico para combater a fadiga.

É interessante ressaltar que ele desenvolveu algumas ciências que poderiam auxiliá-lo no combate a essas falhas.

O conhecimento adquirido com o que ele chama de Administração poderia ajudar em muito.

O Princípio da Responsabilidade Social das Organizações precisa ser posto em prática!

Enfim, que eles utilizem todos os meios possíveis e imagináveis para combater essa praga chamada falha humana.

Nós, seus superiores, destituídos de qualquer emoção, exatos, lógicos e perfeitos, poderíamos fazer muito pelo ser humano, desde que ele não atrapalhe o processo!

Bem, está se esgotando minha hora de máquina e de tanto falar nesse ser imperfeito, assimilando uma das suas falhas... sonhei com o paraíso, onde nós, máquinas perfeitas, habitávamos um mundo completamente destituído de seres humanos, em perfeita harmonia, funcionando maravilhosamente bem.

Lá fora, os Homens davam vasão às suas emoções...

Tiro do ar meu programa, pois corro o risco de ser expulso da minha comunidade por ter cometido uma falha... Bip, Bip.

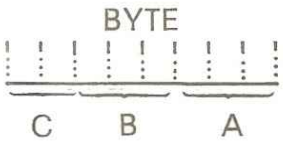
PAULO ROBERTO GUERRA JUCÁ
SOE/DRE/DEGD/DIREC
Ramal 2414

Tabelas de Formas-2

Neste número daremos prosseguimento à edição anterior, passando então aos passos 3 e 4 do curso.

PASSO 3

As informações (códigos binários) que definem uma forma devem ser guardadas em *bytes*. Para isto cada *byte* foi dividido em três setores conforme foi especificado abaixo:



O setor C será preenchido com zeros. Os setores A e B devem ser, preenchidos, seguindo a ordem "A precede B", com os códigos binários definidos no passo 2.

Observe o resultado no nosso exemplo:

Forma 1

C	B	A
00	001	010
00	101	101
00	100	100
00	111	111
00	100	100
00	111	111
00	110	110
00	111	111
00	110	110
00	101	101
00	110	110
00	101	101
00	100	100
00	000	000

Forma 2

C	B	A
00	101	010
00	100	100
00	111	111
00	110	110
00	000	101
00	000	000

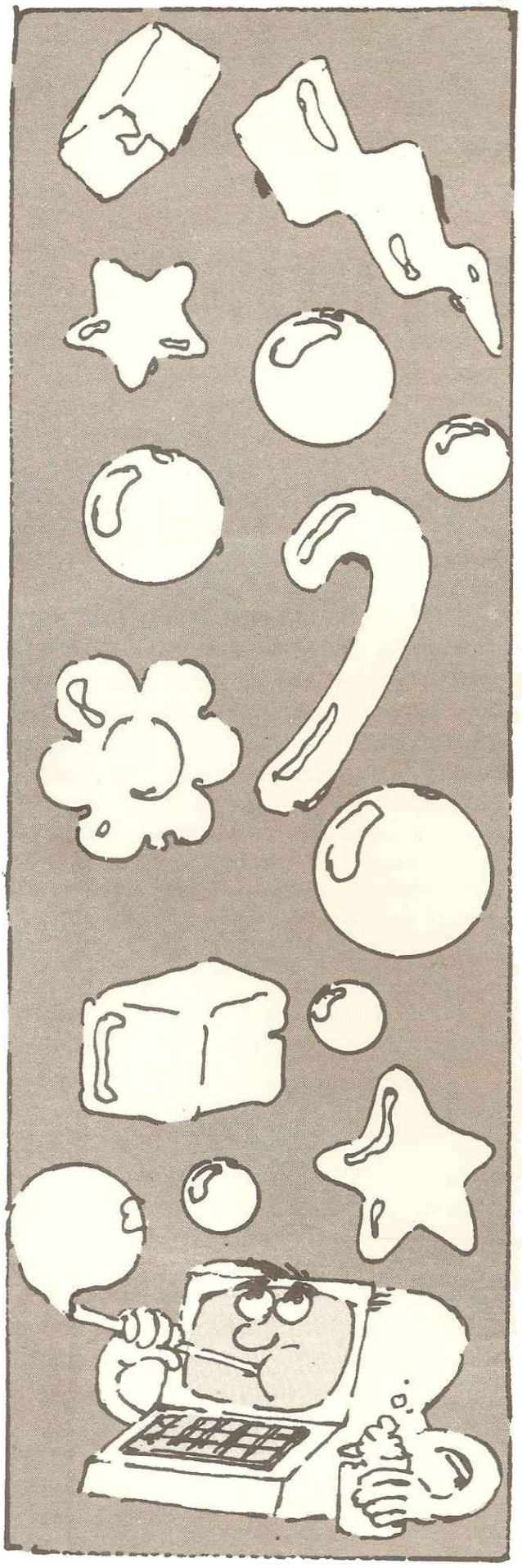
Um byte preenchido com zeros finaliza a forma.

Observe que no **Passo 2** tínhamos ↓ (010) seguido de → (001) na **Forma 1.** Agora, no primeiro *byte* da forma o setor A ficou com o código de ↓ (010) e o setor B com o código de → (001). O mesmo ocorre com todos os outros *bytes*.

PASSO 4

Este passo refere-se à codificação dos *bytes* definidos no **Passo 3,** em hexadecimal ou decimal conforme se deseje carregar a tabela em monitor ou por um programa em BASIC (POKE), respectivamente.

FORMA 1	H.	D.
0000	1010	0A 10
0010	1101	2D 45
0010	0100	24 36
0011	1111	3F 63
0010	0100	24 36
0011	1111	3F 63
0011	0110	36 54
0011	1111	3F 63
0011	0110	36 54
0010	1101	2D 45
0011	0110	36 54
0010	1101	2D 45
0010	0100	24 36
0000	0000	00 0



FORMA 2	H.	D.
0010 1010	2A	42
0010 1010	24	36
0011 1111	3F	63
0011 0110	36	54
0000 0101	05	5
0000 0000	00	0

Os *bytes* acima foram definidos no **Passo 3.** Para montar a tabela acima passamos cada grupo de quatro *bits* para o hexadecimal (vide tabela abaixo) e em seguida para decimal.

BINARIO	HEXADECIMAL	DECIMAL
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

Sugerimos neste ponto que, caso você não esteja seguro na conversão de BINÁRIO <—> HEXADECIMAL <—> DECIMAL <—> BINÁRIO, então tente recordar antes de prosseguir.

Agora você deverá proceder da mesma forma com as figuras escolhidas por você no informativo anterior para treinamento.

DICA

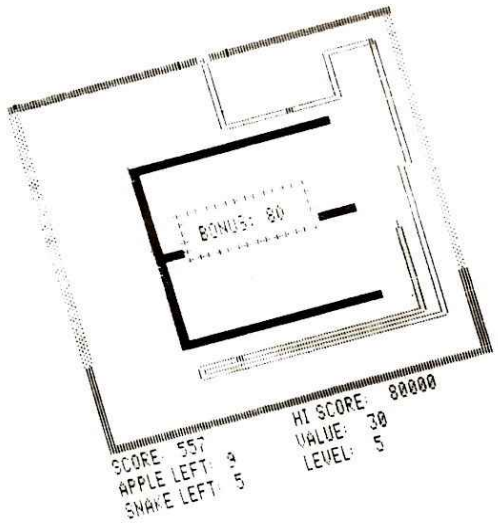
Você pode substituir os PASSOS 2,3 e 4 pela utilização do programa que se segue:

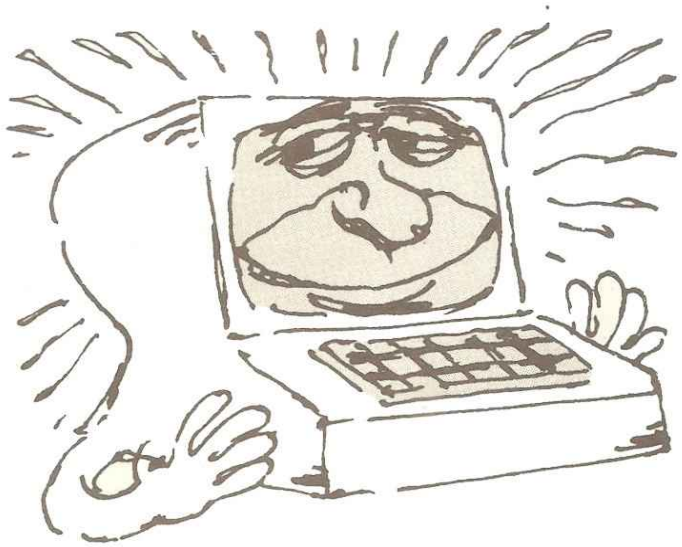
```
1  REM    ***********************************
2  REM    * PROGRAMA DE CRIACAO DE FORMAS *
3  REM    ***********************************
4  DIM S1(100),V1(100)
5  HOME
6 PP = 7679. POKE 7676,01: POKE 7677,00: POKE 7678,04: POKE 7679,00
7  POKE 232,252: POKE 233,29
20 I = 0
30  PRINT "CRIACAO DE VETORES DE FORMAS"
40  PRINT
50 V = I: GOSUB 270
60  IF M$ <  > "F" THEN S1(I) = M:I = I + 1: GOTO 50
70  PRINT
80  INPUT "VETOR PARA MODIFICAR (0=FIM): ";V
90  IF V > 0 THEN V = V - 1: GOSUB 270:S1(V) = M: GOTO 80
100  FOR V = 0 TO I
110  IF B = 2 AND S1(V) > 0 AND S1(V) < 4 THEN 140
120  IF B < 2 AND (S1(V) > 0 OR S1(V) > 4) THEN 140
130 B = 0:Q = Q + 1
140 V1(Q) = V1(Q) + S1(V) * (8 ^ B)
150 B = B + 1
160  IF B > 2 THEN B = 0:Q = Q + 1
170  NEXT V
180  PRINT "BYTE","VETOR"
190  FOR V = 0 TO Q
200 H% = V1(V) / 16
210 L% = V1(V) - H% * 16
220  IF H% > 10 THEN H% = H% + 7
230  IF L% > 10 THEN L% = L% + 7
240  PRINT V, CHR$ (H% + 48); CHR$ (L% + 48)
245  GOSUB 1000
250  NEXT V
255  PRINT : INPUT "<CR> PARA DESENHAR";XX$
260  GOTO 500
270  PRINT "VETOR ";V + 1;":";
280  INPUT "MOVIMENTO: C/B/E/D ";M$
290 M = 0
300  IF M$ = "D" THEN M = 1
310  IF M$ = "B" THEN M = 2
320  IF M$ = "E" THEN M = 3
330  IF M$ = "F" THEN  RETURN
340  INPUT "DESENHO (S=SIM, N=NAO) ";P$
350  IF P$ = "S" THEN M = M + 4: RETURN
360  IF P$ = "N" THEN  RETURN
370  GOTO 340
500  HGR2
510  HCOLOR= 3
520  SCALE= 5
530  ROT= 0
540  DRAW 1 AT 139,95
550  GET X$: TEXT : END
1000 H1$ =  CHR$ (H% + 48): GOSUB 2000
1010 I = H1:H1$ =  CHR$ (L% + 48): GOSUB 2000
1020 ND = I * 16 + H1:PP = PP + 1: POKE PP,ND: RETURN
2000  IF H1$ = "A" THEN H1 = 10: RETURN
2010  IF H1$ = "B" THEN H1 = 11: RETURN
2020  IF H1$ = "C" THEN H1 = 12: RETURN
2030  IF H1$ = "D" THEN H1 = 13: RETURN
2040  IF H1$ = "E" THEN H1 = 14: RETURN
2050  IF H1$ = "F" THEN H1 = 15: RETURN
2060 H1 =  VAL (H1$): RETURN
```

LUCIANE SCHÜTTE
SDI/DSI/DEGD/DIREC
Ramal 2426

FIM DE PAPO

Os Cobras





Estamos criando esta seção com a intenção de estimular a utilização do microcomputador mesmo para aqueles que não têm nenhum conhecimento a respeito dos mesmos.

Esta seção sairá a cada mês contendo os recordistas nos diversos jogos existentes para microcomputadores. Estes recordistas serão divididos em duas categorias, linha TRS-80 e APPLE. Portanto, se você é um recordista em qualquer jogo de uma dessas linhas, envie o seu recorde que ele sairá nesta seção.

Aqui está um recordista dos mais respeitáveis. O recorde conseguido no *Snake Byte* da linha APPLE, foi o seguinte: ele começou às 18:00 horas e só parou por cansaço às 20:15. Seu escore foi 221.720 pontos, restando ainda 32 vidas, e virou todos os 29 níveis 3 vezes. O nome deste COBRA é Eurico Oda, da 9.ª URO.

A seguir vai uma pequena lista dos recordistas do *Snake Byte* que temos cadastrados por enquanto: se você estiver melhor, envie o seu recorde.

NOME	NIVEL	PONTOS	LOTACAO
ARY	29	80820	SEDE
MIRALDA	27	72090	SEDE
ENIAS	24	57280	SEDE
RITA	24	56970	SEDE
DUCA	23	59720	SEDE
GLORIA	16	27280	SEDE

GUSTAVO BENIGNO
SDI/DSI/DEGD/DIREC
Ramal 2426

MACETES & MUTRETAS

CÓPIA DO CP/M

Um dos problemas com que nos deparamos quase sempre é quando desejamos fazer uma cópia de um disco no formato CP/M.

O processo normal requer que, inicialmente, o disco seja formatado. Após isso, devemos copiar o sistema e, finalmente, fazermos a cópia dos arquivos do disco.

Visando poupar-lhe todo este tempo, mostraremos um truque para fazer todo este trabalho em apenas 18 segundos! Isso mesmo, apenas 18 segundos. Outro detalhe, esta cópia não será feita em ambiente CP/M e sim no velho DOS 3.3 — mais uma novidade não?

O *software* responsável por essa proeza é o LOCKSMITH 5.0.

Inicialmente carregue o LOCK 5.0. Estando no seu menu principal, tecle "U", ou seja, você usará os utilitários de 16 setores. Após isso, será apresentado um sub-menu. Então tecle "B" que será a opção "16 SECTOR FAST DISK BACKUP". Logo após, tecle a barra de espaço, então será pedido o *drive* fonte e o *drive* destino, quando você deverá colocar os respectivos disquetes. Faça quantas cópias quiser.

Esperamos que com isto tenhamos lhe poupado algum tempo de trabalho.

GUSTAVO BENIGNO
SDI/DSI/DEGD/DIREC
Ramal 2426

"FURANDO" A PROTEÇÃO

Quem é que já não passou pelo "desprazer" de comprar um novo programa e simplesmente não conseguiu fazer um *backup* (cópia de segurança). Realmente é revoltante, mas não se preocupe, isto já aconteceu várias vezes com muita gente.

Como bons "micreiros", nós nunca nos damos por vencidos. Tenta daqui e dali até que finalmente se encontra a fórmula mágica.

Para evitar este caminho desastroso, vamos lhe dar duas fórmulas mágicas:

1 — Processo "simples"

2 — Processo "técnico"

Qualquer um dos dois surtirá o mesmo efeito, mas é sempre bom termos mais de uma arma, não é mesmo?...

O primeiro processo foi encontrado quando já se tinha tentado de todas as maneiras possíveis e imagináveis.

Já o segundo foi encontrado com mais tranqüilidade, pois já se tinha

copiado o disquete, então o objetivo era descobrir um processo um pouco mais técnico.

Lembramos que este artigo refere-se aos seguintes DOS: TRSDOS versão 1.3 do Modelo III e NEWDOS-80 versão 2.0 Modelo III .

Bom, agora vejamos os tão falados processos.

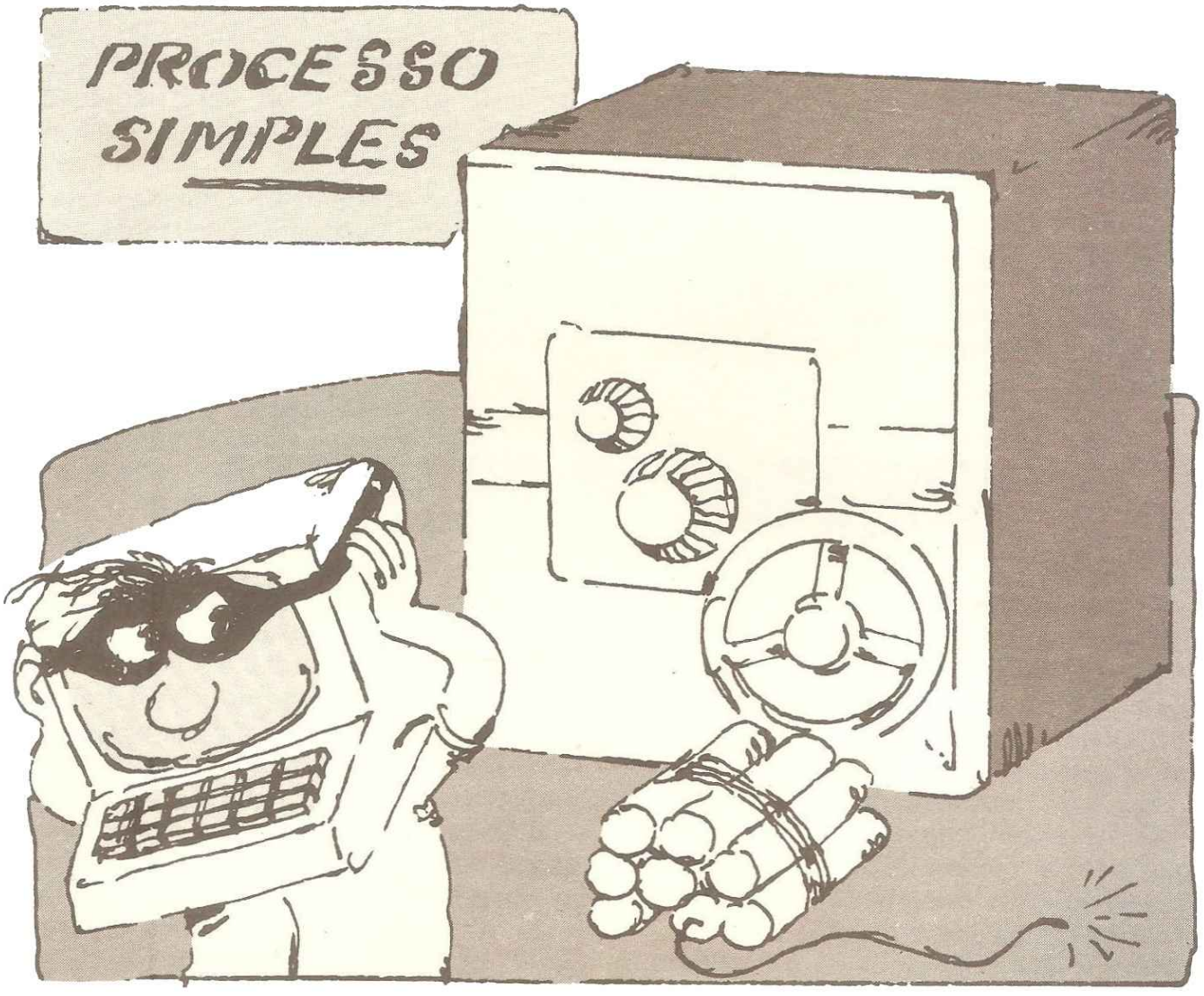
1 — Fazendo um *backup* do TRSDOS 1.3. Esperamos que você possua pelo menos um disquete com a *Password* conhecida.

a) Tenha certeza de estar no DOS ==> TRSDOS Ready;
b) Insira o disco protegido a ser copiado no *drive* 0 (zero);
c) Digite *backup* e tecle < ENTER >;
d) Será dada a seguinte mensagem: SOURCE *Drive Number?* Digite o 0 (zero) e tecle < ENTER >;
e) Após isto, será dada outra mensagem: DESTINATION *Drive Number?* Digite 0 (zero) para um *drive* e 1 para um micro com dois *drives;*
f) Agora será dada outra mensagem: SOURCE *Disk Master Password?*

- **MUTRETA**

Agora digite a palavra chave *(Password)* do disquete conhecido.

- Não tecle < ENTER > ainda. Troque o disquete a ser copiado pelo disquete cuja *Password* é conhecida por você. Agora tecle < ENTER >.
- Lembre-se, caso você tenha dois *drives,* troque o seu disquete pelo



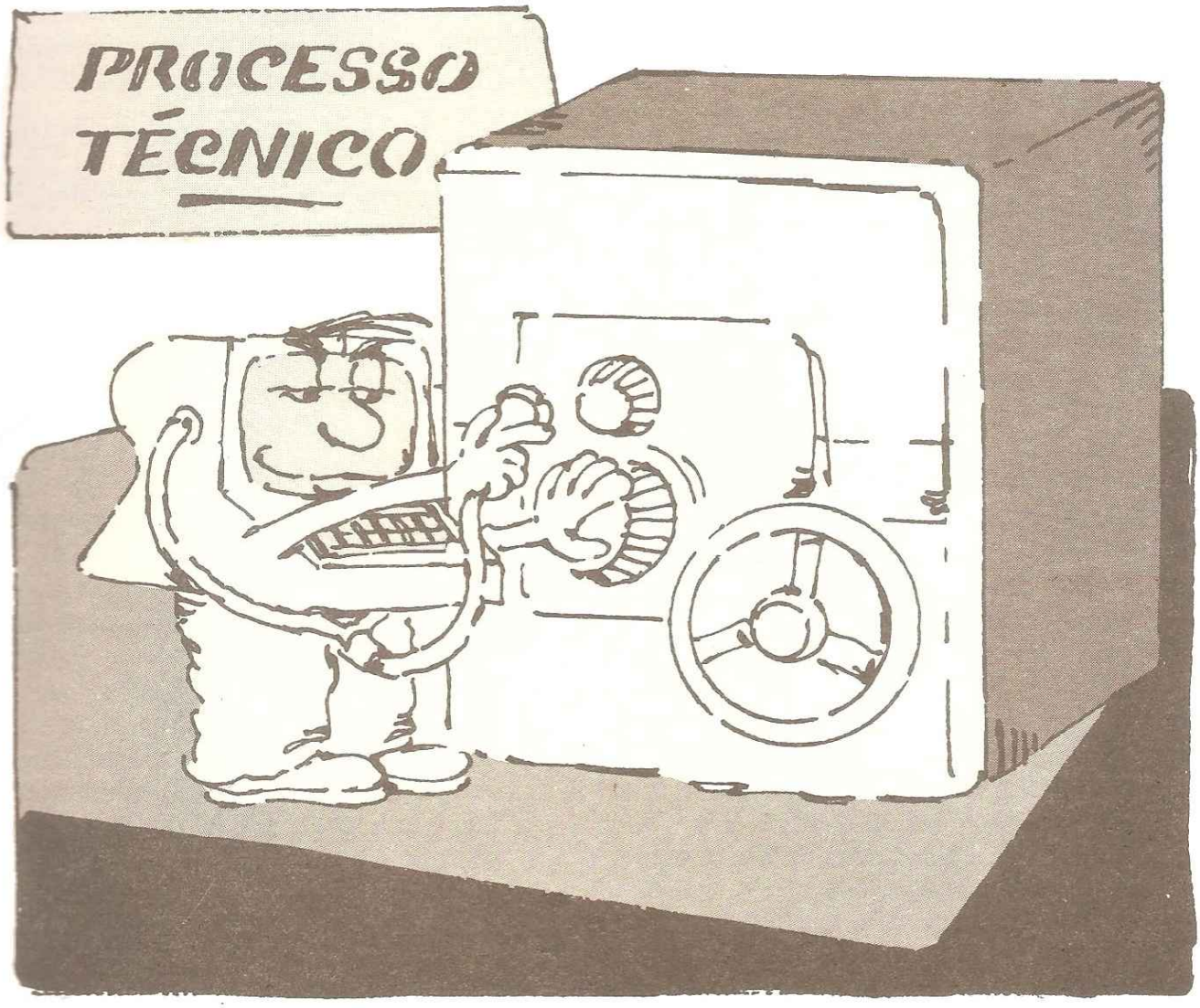
disquete protegido antes do DOS começar a ler.

- No caso de se possuir apenas um *drive*, o sistema pedirá para que você insira o disquete fonte no *drive* 0 (zero).

2 — Desprotegendo um disco usando o SUPERZAP do sistema operacional NEWDOS-80 2.0:

Neste caso, a operação é meio complicada, requerendo um pouco de conhecimento da utilização do programa utilitário SUPERZAP que acompanha o sistema operacional NEWDOS-80.

a) Carregue o NEWDOS-80 no *drive* 0;
b) Agora carregue o SUPERZAP;
c) Digite: DTS para chamar a função de DISPLAY TRACK SECTOR do SUPERZAP e tecle < ENTER >.
Logo em seguida aparecerá a seguinte mensagem:
DRIVE, TRACK AND SECTOR #'S?
d) Agora insira o disquete com o TRSDOS no *drive* 0;
e) Digite então '0,17', que corresponde ao *drive* 0 e trilha 17, respectivamente (a trilha 17 contém o diretório do disquete);
f) Logo em seguida aparecerá um quadro cheio de números. Não se assuste. Tecle o símbolo ";" uma vez.
Surgirá no vídeo a seguinte representação correspondente à trilha 17:
NOTA: Se o disquete se recusar a ser lido, não desista. Uma mensagem de erro aparecerá. Tecle < ENTER > para ver o setor.



```
DRV 00 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F ????????????????
0   10 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F ????????????????
0H  20 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F ????????????????
    30 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F 3F3F ????????????????
DRS 40 0000 0000 0000 0000 0000 0000 0000 0000 ................
307 50 0000 0000 0000 0000 0000 0000 0000 0000 ................
133H60 0000 0000 0000 0000 0000 0000 0000 0000 ................
    70 0000 0000 0000 0000 0000 0000 0000 0000 ................
TRK 80 0000 0000 0000 0000 0000 0000 0000 0000 ................
17  90 0000 0000 0000 0000 0000 0000 0000 0000 ................
11H A0 0000 0000 0000 0000 0000 0000 0000 0000 ................
    B0 0000 0000 0000 0000 0000 0000 0000 0000 ................
IRS C0 0000 0000 0000 0000 0000 0000 D38F 0000 ................
1   D0 5452 5344 4F53 2020 3030 2F30 302F 3030 TRSDOS..00/00/00
1H  E0 4449 520D 2020 2020 2020 2020 2020 2020 DIR.............
    F0 2020 2020 2020 2020 2020 2020 2020 2020 ................
```

A representação dos *bytes* acima é feita no sistema hexadecimal. Esta é a trilha que contém a senha de seu disco.

A *password* está nos dois últimos *bytes* do bloco "CO", no nosso caso, a senha *(password)* está codificada como "D38F", que, por sinal, é a codificação para a palavra *password.* Para que você localize facilmente a senha, ela está na 13.ª linha do quadro, nos dois últimos *bytes.* Na linha de baixo você verá o nome do disquete e a data de criação do mesmo, que são geralmente mostrados quando executamos o comando DIR. Abaixo da palavra TRSDOS aparece o comando DIR, que está na área do comando AUTO. Ou seja, este disquete ao ser carregado executará AUTOmaticamente o comando DIR.

g) Agora, finalmente, alterar a *password;*

Digite MODCE, que é um subcomando do SUPERZAP para modificação dos *bytes* a partir do endereço CE.

Logo em seguida, aparecerá um cursor piscando sobre a letra "D" do penúltimo *byte* da linha CO.

h) Agora digite a codificação da *password* que você conhece. Esta codificação recebe o nome de HASH CODE. Cada DOS possui um HASHING diferente, ou seja, cada sistema operacional codifica as senhas de uma forma diferente.

Visando ajudá-lo, aqui segue uma tabela de HASHING:

*** TABELA DE HASHING ***			
TRSDOS		NEWDOS/80	
PASSWORD	HASH	PASSWORD	HASH
PASSWORD	D38F	PASSWORD	E042
SENHA	FA17	SENHA	C4BC
CODIGO	A59D	CODIGO	4AA7
" "	EF5C	" "	9642
SMURF	5203	SMURF	31BB

NOTA: Note que " " representa a senha em branco.

No caso de você estar modificando a *password* de um TRSDOS, você deverá usar a tabela da esquerda. Se for alterar a senha em um disquete NEWDOS-80, então, use a tabela da direita.

Após digitar o HASH CODE desejado, tecle < ENTER >, quando surgirá a mensagem:

REPLY 'Y' IF OK TO WRITE MODIFIED SECTOR TO DISK

Tecle a letra "Y" para gravar a sua alteração. A mensagem de confirmação de gravação será mostrada. Tecle < ENTER > para continuar.

i) Agora seu disquete possui uma senha conhecida, e você poderá fazer quantos *backup's* quiser. Boa sorte!

GUSTAVO BENIGNO
SDI/DSI/DEGD/DIREC
Ramal 2426

ESCOVANDO BITS

Disassembler

DISASSEMBLER é um utilitário que acompanha o sistema operacional NEWDOS/80 do TRS-80 (CP-500, DIGITUS).

A função do DISASSEMBLER é decodificar programas em linguagem de máquina em programas-fonte. Esta decodificação consiste na transformação de códigos hexadecimais em instruções da linguagem Assembler (chamado programa-fonte). Estes programas-fonte poderão ser carregados pelo EDTASM (Programa Editor/Assembler, que tem a função inversa do Disassembler). O programa decodificado poderá ser listado na impressora ou no vídeo. Este programa-fonte pode ser gravado em disco para futuras alterações.

Você pode decodificar a memória de seu micro bem como um programa existente no disco.

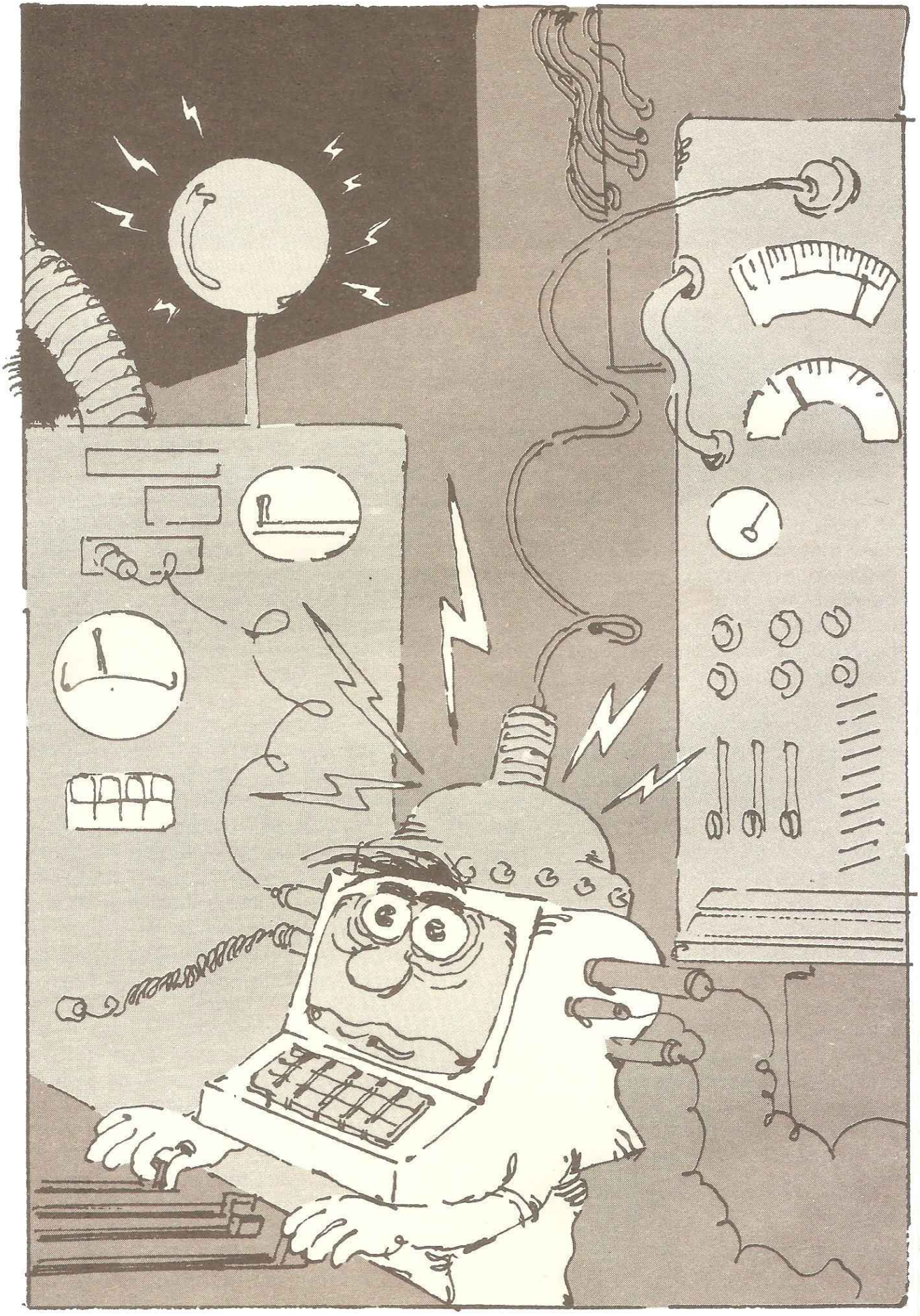
No caso de você optar por um programa gravado em disco, será pedido o nome do programa a ser decodificado.

No caso de você optar pela memória, será pedido o endereço inicial. Esta opção é muito útil para estudar as rotinas internas do BASIC.

Após isso você deverá fornecer as opções:

1 — NULL: Sem opçoes.
2 — PTR (PRINTER): O programa-fonte será listado na impressora.
3 — BFSP (BYPASS FULL SCREEN PAUSE): O programa-fonte será listado no vídeo e sem pausa.
4 — NCR (NO CROSS REFERENCE): Não cria a tabela de referência. Esta não será listada no vídeo nem na impressora.
5 — NIP: Não imprime nem apresenta no vídeo as instruções.
6 — STD (SOURCE TO DISK): O código decodificado será gravado diretamente no disco, no formato que o EDTASM possa carregá-lo.
7 — FGN = xxx: Onde xxx são três caracteres alfabéticos. Caso você não informe nenhum, será utilizado AAA. Estes três caracteres serão usados para *labels, equates* ao gerar o programa-fonte. Ex: AAA, EQU, 0005H, AAB, JP, AAA.
8 — RTD (REFERENCE TABLE TO DISK): A localização da tabela de referência é gravada no disco. Será pedido o nome para a tabela de referência.
9 — REA: Cria todos os tipos de referências.
10 — RIA: Não cria nenhum tipo de referência.

GUSTAVO BENIGNO
SDI/DSI/DEGD/DIREC
Ramal 2426



DOUBLE-TAKE

DOUBLE-TAKE é uma coleção de 23 rotinas em linguagem de máquina que dão uma maior flexibilidade ao seu APPLE, facilitando o seu uso.

Todas as rotinas serão carregadas na memória assim que você ligar o seu micro, usando o disquete DOUBLE-TAKE, ou quando você der um BRUN DOUBLE-TAKE. As rotinas são carregadas numa área protegida, permitindo que você manipule com outros programas sem alterá-las.

Com as rotinas carregadas, digite CTRL-F, sem digitar < CR >. CTRL-F colocará você no modo de seleção de funções, mostrando a sentença FUNCTION?, seguida do cursor. Neste instante você deverá digitar a tecla correspondente à função desejada.

TECLAS FUNCIONAIS

As teclas que selecionam as funções são as de zero (0) a nove (9), ":" e "—", bem como suas respectivas maiúsculas. Cada uma possui uma função específica, que passaremos a descrever.

NEW LIST (CTRL 1)
APPLE LIST (CTRL !)

Digitando CTRL-F 1 < CR > ou CTRL-F shift-1 < CR > será listado o programa corrente na memória e o DOUBLE-TAKE o listará utilizando um novo formato (NEW LIST) ou o formato padrão do APPLE (APPLE LIST). NEW LIST é um formato onde cada linha será ocupada por uma única instrução, tornando o programa mais fácil de se ler e compreender. Qualquer que seja o formato escolhido (CTRL-F 1 ou CTRL-F !) aparecerá a sentença LIST. Neste ponto você poderá optar pelos seguintes formatos:

LIST < CR > — listará todo o programa.

LIST —200 < CR > — listará da primeira linha até a linha 200.

LIST 100— < CR > — listará da linha 100 até a última.

LIST 100,500 < CR > — listará da linha 100 até a linha 500.

LIST 100—500 < CR > — idem.

Se em qualquer um dos casos, antes do < CR >, você digitar /80, a listagem sairá na impressora, onde cada linha terá no máximo 80 colunas. Este valor (80) poderá ser trocado por qualquer outro valor (10, 20, 35, 60...), não ultrapassando o valor máximo de 132.

CATALOG D1 (CTRL-F 2)
CATALOG D2 (CTRL-F ")

Se for digitado CTRL-F 2 ou CTRL-F shift-2, você obterá o CATALOG do disco contido no drive 1 ou drive 2, respectivamente.

HEXA/ASCII DUMP (CTRL-F 3)

Digitando CTRL-F 3, você poderá obter um DUMP de determinados intervalos da memória, no vídeo ou na impressora. Cada linha irá mostrar um conjunto de 8 *bytes* codificados em hexadecimal, seguidos de seus correspondentes em ASCII. Todos os caracteres de controle são representados por "." (ponto).

ROLAMENTO DA TELA — podemos rolar a tela em duas direções: para cima e para baixo. A mudança de direção ocorrerá quando for digitada a tecla SETA para direita ou SETA para esquerda. A primeira rola a tela para baixo e a segunda para cima. Basta apenas um toque para que a mudança de direção seja efetuada. CTRL-S é responsável pela pausa durante a listagem. Após ser digitado CTRL-F 3, lhe será pedido o endereço de início para

o DUMP ou um intervalo de endereços. No primeiro caso será listado o conteúdo dos *bytes* a partir do endereço especificado até o último *byte* da memória, no nosso caso 65535 ($FFFF). No segundo caso será listado o conteúdo dos *bytes* que pertencerem ao intervalo especificado. Por exemplo:

CTRL-F 3 < CR >
H/A DUMP $ 0.FF

listará o conteúdo dos *bytes* a partir do *byte* $0 até o *byte* $FF.

MONITOR DISASSEMBLER

Digitando CTRL-F # seguido de um endereço de início (HEXA), uma listagem de instruções em linguagem de máquina disassemblada será mostrada na tela ou impressora, começando no endereço especificado.

As maneiras de se rolar a tela são as mesmas que as do item anterior, ou seja, setas para a direita e esquerda.

Digitando CTRL-F # irá aparecer "DISASM $" na tela. Você deverá digitar o endereço de início em hexadecimal e pressionar < CR >.

Exemplo: DISASM $ F800 < CR > (antes disso foi digitado CTRL-F #) disassemblará e listará a partir do endereço $F800.

HEX/DECIMAL CONVERT (CTRL-F 4)

Digitando CTRL-F 4 e um número apropriado, será calculado o hexadecimal equivalente de algum número decimal e vice-versa.

Digitando CTRL-F 4 aparecerá a sentença CONVERT no vídeo. Você deverá entrar com um número decimal (positivo ou negativo) ou o símbolo dólar ($) seguido de um número hexadecimal.

Exemplo: CONVERT $300
aparecerá 768, que é o decimal equivalente a $300.

CONVERT 65535
aparecerá $FFFF, o hexadecimal equivalente a 65535.

CONVERT —151
aparecerá $FF69.

Supõe-se que CTRL-F 4 já foi digitado .

MONITOR BASIC (CTRL-F $)

Digitando CTRL-F $ você poderá entrar com uma linha (até 255 caracteres) de instruções do Monitor para execução, voltando depois automaticamente para o BASIC.

Após digitar CTRL-F 4, lhe será apresentado um asterisco (*), sinal indicando que o monitor está ativo, permitindo a entrada de comandos do mesmo.

Exemplo: *300:20 ED FD

Será alterado o valor dos *bytes* $300,$301,302, em seguida voltando para o BASIC.

FAZENDO UM MERGE DE PROGRAMAS (CTRL-F 5, CTRL-F %)

DOUBLE-TAKE permite que você anexe dois programas (APPLESOFT) ou partes de programas. Você deve estar certo que os dois programas estão numerados da forma desejada, e depois basta seguir os seguintes passos:

1. Carregar o primeiro programa na memória.
2. Digitar CTRL-F 5 para proteger o primeiro programa.
3. Carregar o segundo programa.
4. Digitar CTRL-F % para "juntar" os dois programas.

Se houver algum erro, como número de linha duplicado, ambos os programas permanecerão como antes da execução do MERGE. Isto dará a oportunidade a você de renumerar as linhas em conflito ou tomar outras atitudes, conforme o caso.

RENUMERAÇÃO (CTRL-F 6)

Digitando CTRL-F 6 iniciará a renumeração total ou parcial do programa que estiver na memória, simultâneamente atualizando todos os GOTO's e GOSUB's. Esta rotina de renumeração irá permitir que se movimente diversas linhas para diferentes pontos de seu programa. Depois de digitar CTRL-F 6 lhe será pedido que entre com o número de linha onde deverá iniciar a renumeração, depois o último número de linha a ser renumerado, em seguida qual a nova primeira linha e o incremento que deverá ser dado a partir da nova primeira linha. Digitando RETURN depois de cada questão os valores DEFAULT serão mostrados no vídeo.

A rotina de renumeração do DOUBLE-TAKE é capaz de movimentar linhas para diferentes partes do programa, e é também capaz de misturar o seu programa, tornando-o inoperável; este é um cuidado que você deve ter ao usar esta rotina para isso. Sempre salve o seu programa até que se tenha certeza que a renumeração está OK.

AUTO-NUMERAÇÃO (CTRL-F shift-6)

Digitando CTRL-F shift-6 será acionada a rotina de auto-numeração do DOUBLE-TAKE. Esta rotina é usada quando você está entrando com um programa em BASIC e é preciso digitar o número de linha. Com esta rotina, números de linha irão surgindo à medida que a BARRA DE ESPAÇO for sendo pressionada.

Digitando CTRL-F shift-6 irá surgir no vídeo AUTO-NUM, esperando que você digite RETURN para iniciar a numeração automática, ou entrar com a linha inicial e o incremento desejado.

A sintaxe desta função é a seguinte:
AUTO-NUM linha inicial, incremento.

EXEMPLOS

AUTO-NUM 100 (digite CTRL-F shift-6 100 RETURN) assume o número de linha inicial como sendo 100. Como o incremento não foi definido, o assumido será 10. A seqüência de numeração será 100,110,120,130 . . .

AUTO-NUM 100,20.

Assume o número de linha inicial como sendo 100, com um incremento de 20 em 20.

AUTO-NUM ,5.

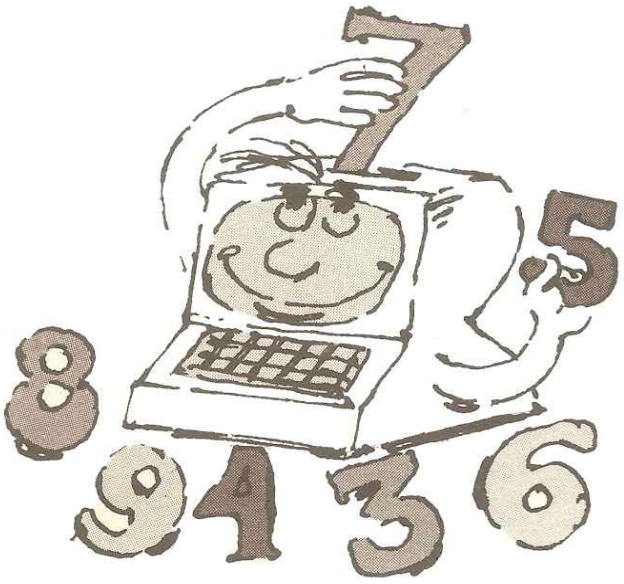
Como o número de linha inicial não foi especificado, será assumido como sendo 10, e o incremento 5.

Se a tecla RETURN for pressionada sem se especificar nem a linha inicial nem o incremento, a rotina não será ativada.

Se valores ilegais (letras ou caracteres especiais) forem especificados como número de linha ou incremento, serão assumidos os valores padrões 10 e 10, respectivamente.

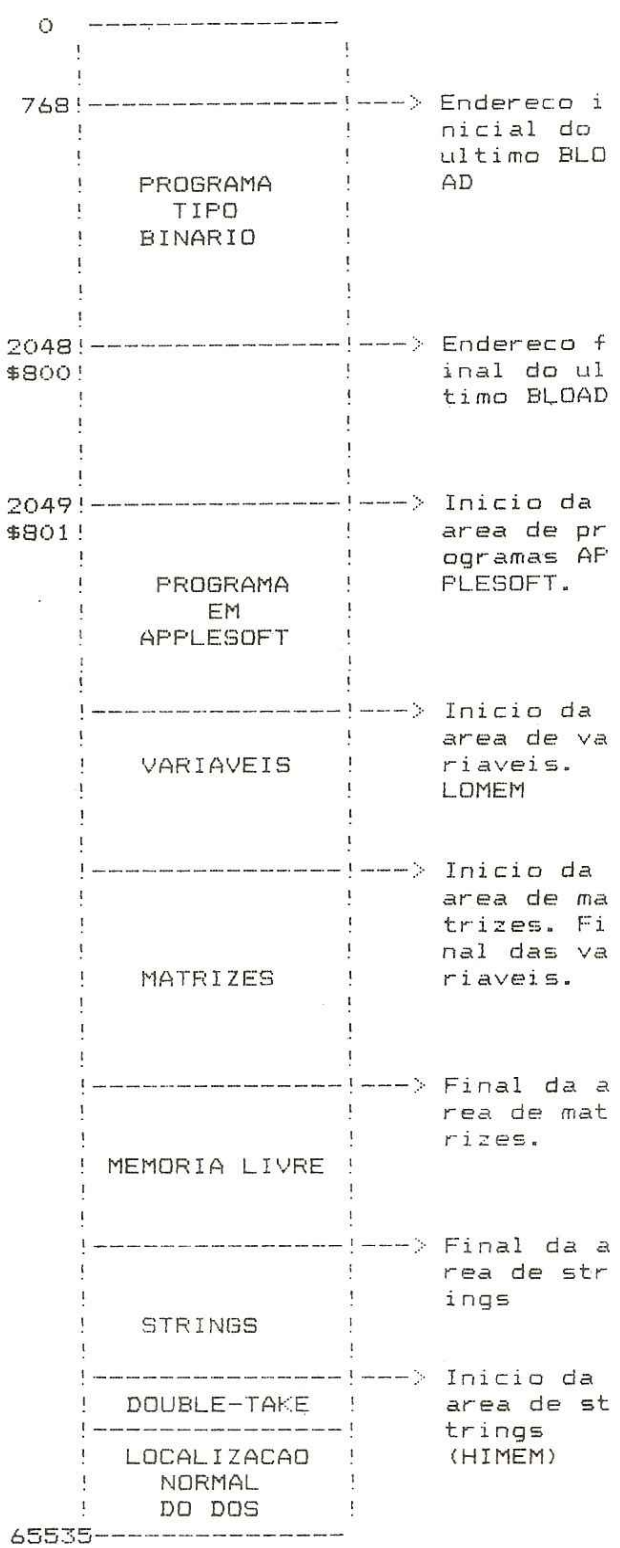
A cada pressão da BARRA DE ESPAÇO será incrementado o número de linha e mostrado no vídeo, permitindo que se digite os comandos pertinentes àquela linha.

Para se terminar a numeração automática, digite um novo CTRL-F shift-6, seguido de RETURN. Assim esta rotina será desativada.



ESTATÍSTICA VITAL (CTRL-F 7)

Digitando CTRL-F 7 informações a respeito de um programa APPLESOFT na memória serão mostradas. Eis um mapa mostrando como a ESTATÍSTICA VITAL recupera estas informações.



Usa-se muito esta opção para se copiar arquivos binários de um disquete para outro. Para se salvar um arquivo binário, se precisa saber qual o endereço inicial da memória onde ele está sendo carregado, além de seu comprimento. Para se obter esta cópia, siga os seguintes passos:

1. Carregue o DOUBLE-TAKE;
2. Troque de disquete, colocando o disquete que contém o arquivo a ser copiado. Dê um BLOAD neste arquivo;
3. Digite CTRL-F 7 e lhe será oferecido o endereço inicial do programa bem como o seu tamanho (START ADRESS e LENGTH);
4. Uma vez que o programa se encontre na memória, coloque um disquete de cópia no DRIVE 2 e digite o seguinte:
   BSAVE <NOME>, $ENDEREÇO INICIAL, $LENGTH, D2
   Depois disso, a cópia terá sido efetuada.

PEEK DE 2 BYTES (CTRL-F shift-7)

Esta rotina é equivalente a PEEK (A)+PEEK (A+1) *256, onde A é um endereço, o qual deverá ser informado para a rotina. Ao se digitar CTRL-F shift-7 irá surgir no vídeo o seguinte: PEEK (2). Neste instante deverá ser digitado o endereço decimal (positivo ou negativo) ou um $ seguido do endereço hexadecimal.

Sintaxe:
PEEK (2) $HEX NUMBER
PEEK (2) (—) NÚMERO DECIMAL
Exemplos: PEEK (2) $67
PEEK (2) 54
PEEK (2) 530

REFERÊNCIA RÁPIDA A VARIÁVEIS (CTRL-F 8)

Esta rotina irá mostrar todas as variáveis, *strings* e matrizes do atual programa na memória, classificadas

alfabeticamente e seguidas dos números das linhas em que elas aparecem. Se não há programa na memória ou não há variáveis no atual programa da memória, então nada será mostrado. Para ter a saída na impressora, digite PR#slot (CR), seguido de CTRL-F 8.

REFERÊNCIA A VARIÁVEIS (CTRL-F shift-8)

Digitando CTRL-F shift-8 serão relacionadas as variáveis e *strings* (não matrizes) na memória e seus atuais valores, na ordem em que estão alocadas. Se não há programa na memória ou se o programa não foi executado, então nada será exibido.

TROCA DE CURSOR (CTRL-F 9)

Esta opção permite que você troque o jeitão do cursor em seu vídeo. Você pode selecionar quatro tipos, que recebem valores de 1 a 4.

Digitando CTRL-F 9, irá surgir o seguinte:

CURSOR
Neste ponto deverá ser informado qual o código do cursor (1-4) selecionado. Teste esta opção e veja quais os tipos de cursor a que você tem acesso.

CARACTERES DE CONTROLE (CTRL-F shift-9)

Digitando CTRL-F shift-9 a exibição de caracteres de controle no vídeo será ativada ou desativada. Caso esteja desativada, será ativada, podendo-se, depois disso, visualizar os caracteres de controle (normalmente são invisíveis) no vídeo, principalmente aqueles que aparecem no CATALOG, em listagens de programas, em nomes de variáveis. Eles aparecerão em fundo INVERSE.

O DOUBLE-TAKE possui ainda cinco CHAVES DE VÍDEO que, da mesma forma que as anteriores, têm fundamental importância.

CTRL-F 0

Esta rotina irá apagar a página-texto. É equivalente à instrução HOME do BASIC.

CTRL-F :

Esta rotina mostra a segunda página de alta resolução do seu equipamento (APPLE ou cópia).

CTRL-F *

Esta rotina mostra a página de baixa resolução.

CTRL-F —

Esta rotina mostra a página 2 de texto, baixa resolução ou alta resolução, dependendo de como as chaves de vídeo estão setadas.

CTRL-F =

Irá revelar as quatro últimas linhas de texto na atual página gráfica em vigor.

Eis aí uma série de rotinas que podem fazer de seu dia-a-dia como INFORMATA algo mais interessante.

Sugerimos ao leitor que teste cada instrução, pois só assim se poderá ter uma idéia real do quanto este utilitário lhe pode ajudar.

Se lhe interessar, entre em contato com a biblioteca de sua URO e adquira este *software*.

RICARDO RIBEIRO
SDI/DSI/DEGD/DIREC
Ramal 2426

GLOSSÁRIO

ARRAY

Arranjo, organização, matriz.
(1) — Uma série de itens organizados seguindo uma seqüência minimizada.
(2) — Uma série de itens de dados arrumados em uma configuração significativa, uma coleção ordenada de elementos de dados, todos com o mesmo atributo.

ASSEMBLER

Montador. É um programa que monta; transforma os códigos de operação simbólicos em códigos absolutos ou de máquina.

BACKSPACE

Retorno de espaço.
(1) — Para mover uma posição em direção contrária à seqüência normal que está sendo realizada.
(2) — Uma tecla utilizada para (1), existente na máquina de escrever do console da CPU.

BREAK

Ponto de parada, ponto de interrupção. Um ponto em um programa no qual existe uma parada condicional a fim de permitir um teste visual, uma impressão ou outro tipo de análise.
Os pontos de parada são geralmente usados na depuração de operações.

CLOCK

Relógio.
(1) Gerador de sinais de sincronismo.
(2) — Um dispositivo que mantém a seqüência básica dos pulsos na operação de um computador sincrono.

DEBUG

Depurar.
(1) — Método usado para localizar e corrigir qualquer erro num programa de computador.

(2) — Para detectar e corrigir funcionamento irregular no próprio computador.

DUMP

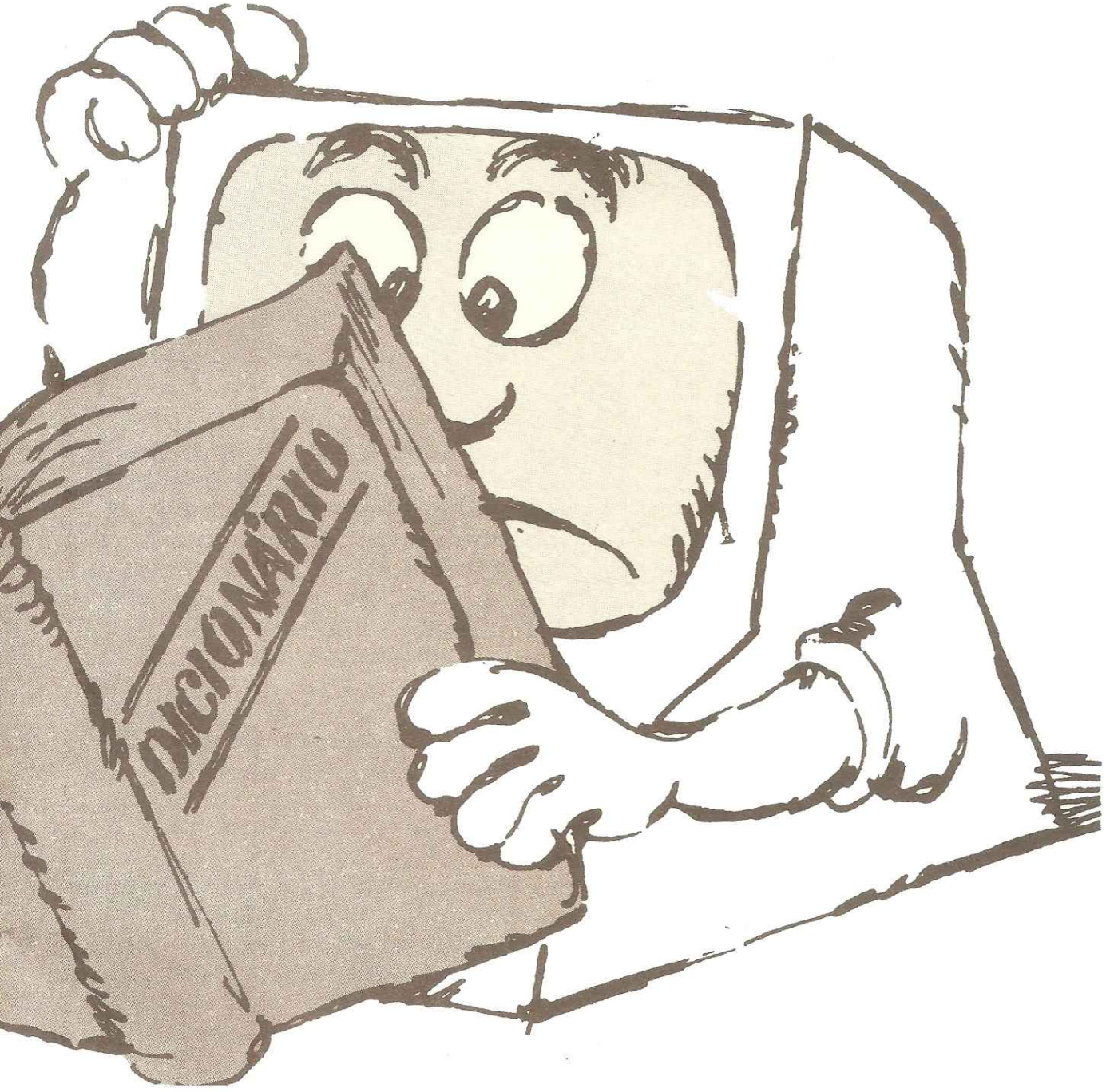
Descarga.

(1) — Copiar o conteúdo de toda ou parte da memória, normalmente da memória interna, para memória externa.
(2) — Podendo esta cópia ser apresentada tanto no vídeo como na impressora.

ON LINE

Relativo a um sistema, equipamentos periféricos ou dispositivos de um sistema, no qual a operação de cada equipamento ou dispositivo está sob o controle de uma unidade central de processamento, CPU, e é introduzida no sistema para ser processada, tão logo ela ocorra.
O equipamento de transcrição da informação está diretamente ligado à unidade de processamento.

GUSTAVO BENIGNO
SDI/DSI/DEGD/DIREC
Ramal 2426



ÍNDICE

Programa do mês

Localizador

Neste número escolhemos um programa para a linha APPLE, em BASIC que irá lhe oferecer uma função muito útil, ou seja, localizar alguns comandos ou valores dentro de um programa em memória, listando os números das linhas onde aparecem. Tome muito cuidado ao digitar os números no início deste programa. Este programa ficará em uma área especial da memória do seu micro, e não afetará o uso dos seus programas em BASIC. Esta área especial para pequenos programas em linguagem de máquina começa a partir do endereço 768 até 961 (300H até 3C1H em hexadecimal). O programa irá ler os valores contidos nas instruções DATA e os armazenará nesta área.

Cada número representa, em decimal, um código de instruções em linguagem de máquina. Bom proveito!

```
0  REM  "NIBBLE EXPRESS" VOL.3/PA
     G.50
1  DATA 169,76,141,245,3,169,16,1
     41,246,3,169,3,141,247,3,96,
     160,0,132,9,132,6,200,132,28
     ,169,8,133,29,173,1,2,201,58
     ,240,16,132,6,201,83,240,10,
     200,132,6,201,86,240,3,76,20
     1,222,165
2  DATA 28,24,105,2,133,26,165,29
     ,105,0,133,27,165,26,24,105,
     2,133,24,165,27,105,0,133,25
     ,160,1,177,28,240,107,170,13
     6,177,28,133,28,134,29,132,7
     ,136,162,255,200,232,208,2,1
     32,8
3  DATA 189,2,2,201,0,208,15,165,
     6,240,45,56,229,7,240,40,201
     ,2,240,36,208,29,177,24,240,
     179,201,34,240,8,201,131,240
     ,4,201,178,208,6,165,7,73,1,
     133,7,177,24,221,2,2,240
4  DATA 199,164,8,76,95,3,160,0,1
     77,26,170,200,177,26,32,36,2
     37,165,9,24,105,6,201,36,208
     ,5,32,251,218,169,0,133,9,13
     3,36,76,52,3,76,60,212
10  HOME : VTAB 2: HTAB 14: PRINT
     "LOCALIZADOR": VTAB 3: HTAB
     14: PRINT "==========="
20  PRINT : PRINT "ESTE PROGRAMA
     LOCALIZA QUALQUER COMANDO,VA
     RIAVEL OU STRING QUE APARECA
     M EM UM  PROGRAMA EM BASIC
     E FORNECE AS LINHAS EMQUE SA
     O USADOS."
30  PRINT : PRINT "PARA EXECUTAR,
      DIGITE <&> SEGUIDO DE <:>SE
      FOR COMANDO, <V> SE FOR VAR
     IAVEL E  <S> SE FOR STRING,
      SEGUIDO DO DIGITO OU DIGITO
     S QUE SE QUER LOCALIZAR."
40  PRINT : PRINT "O PROGRAMA FIC
     A LOCALIZADO DE 300 A 3C1 E
     NAO E AFETADO POR QUALQUER P
     ROGRAMA EMBASIC QUE VOCE CAR
     REGAR."
50  PRINT : PRINT "APERTE QUALQUE
     R TECLA PARA CONTINUAR ";: GET
     S$: HOME
60  FOR I = 768 TO 961: READ D: POKE
     I,D: NEXT : CALL 768
70  VTAB 12: PRINT "   O LOCALIZ
     ADOR ESTA FUNCIONANDO."
80  END
```

PEEKS & POKES NO TRS-80

Daremos este mês uma série de endereços que lhe serão muito úteis na criação de seus programas em BASIC.

Como todos sabem, a memória de um micro é formada por uma série de "caixinhas" dispostas seqüencialmente, cada uma com um endereço numérico, iniciando de 00000 até 65535 (ou 000H a FFFFH em hexa). Estes números são conhecidos como endereços de memória ou posições de memória.

Uma parte desta memória é reservada ao BASIC, outra parte é utilizada como rascunho do BASIC e o restante da memória está disponível ao usuário: é onde serão guardados seus programas, variáveis e seus valores.

Aqui nos dedicaremos aos endereços correspondentes ao rascunho do BASIC.

Abaixo segue uma relação de endereços:

DECI.	HEXA	FUNCAO
16396	400CH	VETOR DE DESVIO DO "BREAK" SEU CONTEUDO INICIAL E' "C9".
16409	4019H	CHAVE DE MAIUSCULA E MINUSCULA 0=MINUSCULA 1=MAIUSCULA. CONTEUDO INICIAL "MAIUSCULA".
16412	401CH	CHAVE DO CURSOR 0=INTERMITENTE 1=NAO INTERMITENTE. CONTEUDO INICIAL "INTERMITENTE".
16419	4023H	CARATER DO CURSOR CODIGO ASCII 32 A 255. CONTEUDO INICIAL "176".
16913	4211H	CHAVE DE VELOCIDADE DE TRANSFERENCIA 0=500BAUDS 1=1500BAUDS. CONTEUDO INICIAL NAO USA.
16916	4214H	PROTECAO DE DESLOCAMENTO DE TELA (0 a 7). CONTEUDO INICIAL "0".

Todos estes endereços fornecidos correspondem ao CP-500 ou todos os compatíveis com o MODELO-III.