SISTEMA DE OPERAÇÃO EM DISCO

DOS 400

EDITELE

SISTEMA DE OPERAÇÃO EM DISCO

DOS 400

EDITELE

EDITELE

Editora Técnica Eletrônica Ltda.
Rua Casa do Ator, 1060
04546 — São Paulo — SP — Brasil
Cx. Postal 30141

1.ª EDIÇÃO
1.ª IMPRESSÃO

SUMÁRIO

CAPÍTULO 1

apresentação

INSTALAÇÃO

Antes de instalar a Unidade de Disco, você deve conectar seu computador à TV. Esse procedimento está descrito em detalhes no livro que acompanha o seu CP 400. Uma vez tomada essa providência, você já poderá realizar a conexão da Unidade de Disco da seguinte forma:

ATENÇÃO: Apenas conecte o computador ao aparelho de TV. Não o ligue ainda. Isso só poderá ser feito após conectada a Unidade de Disco.

CONEXÃO DA UNIDADE DE DISCO

É extremamente simples instalar esse periférico ao seu computador (veja na figura 1). A unidade, que chamaremos CP 450, dispõe de um cabo flexível com dois conectores nas extremidades muito parecidos com cartuchos. (A instalação do cartucho está descrita no manual do CP 400.) A grande diferença é que o cabo flexível deve ficar para fora da gaveta. Mas você pode fechar normalmente a tampa, pois esta, quando fechada, permite a acomodação do cabo, que deve passar por baixo de seu computador, indo até o CP 450 (veja a figura 2).

Agora você já tem sua Unidade de Disco instalada e pode utilizar toda a potência de computação que isso representa.

Se a sua Unidade de Disco CP 450 possuir dois "drives" (acionadores de disco), o da esquerda será chamado *drive 0*, enquanto o da direita, *drive 1*. Se tiver apenas um, este será o *drive 0*.

Figura 1

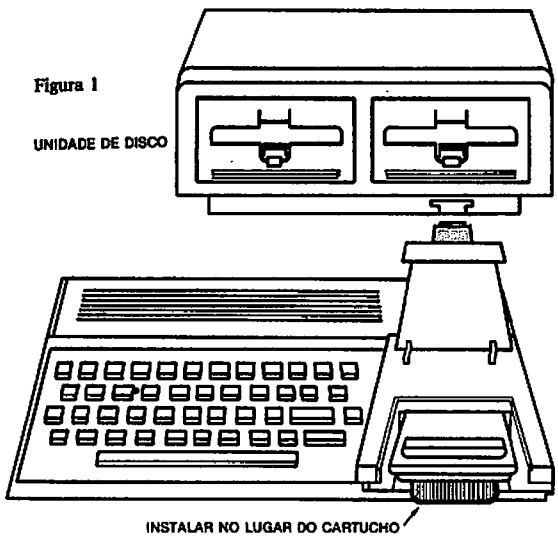
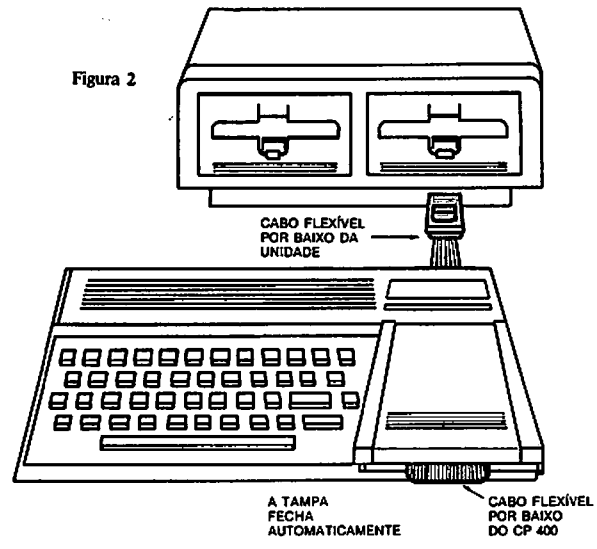


Figura 2



ACIONAMENTO

Como o seu sistema de computador agora está composto por várias partes distintas, você vai precisar mexer em vários botões para ligar todo o sistema:

* Ligue o aparelho de TV.
* Selecione o canal 3 ou 4 (tanto no aparelho como também na chave atrás de seu CP 400).
* Ligue o computador.
* Ligue a Unidade de Disco. O interruptor fica na parte de trás do aparelho.

Se você ligou todos os botões, deve aparecer a seguinte mensagem na tela da TV:

```
BASIC     DISCO v.r
CP 400
PROLOGICA
OK
```

v.r são dois números que indicam qual a versão (V) que você dispõe e qual a revisão (r).

Se isso não ocorrer, desligue todo o sistema, verifique todas as ligações, e tente acioná-lo outra vez.

E AGORA, O MEU DISCO PREFERIDO

Se você pretende estudar todas as possibilidades da sua nova Unidade de Disco, use o disco não formatado que vem com o equipamento. Um disco não formatado é um disco em branco, sem informação nenhuma gravada nele (é o mesmo que uma fita "virgem"). Utilize para as explicações a seguir o *drive 0*. Se você tiver pressa em utilizar seu equipamento com algum programa já pronto, utilize o disco com o "aplicativo" que você quer mas, antes, termine de ler este capítulo para que possa saber conservar os seus discos por bastante tempo.

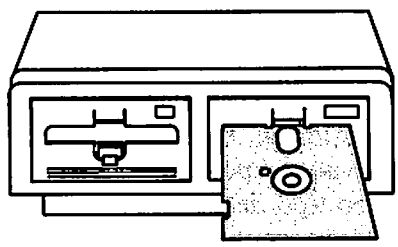
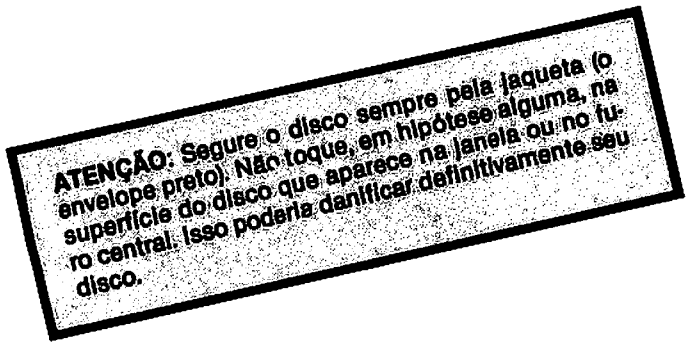


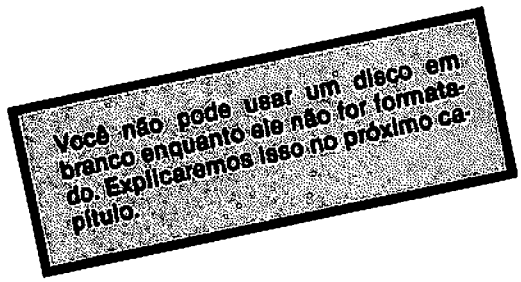
Figura 3

* Abra a porta do drive (pressionando o botão de liberação logo abaixo da mesma).
* Posicione o disco com o chanfro para a esquerda e a etiqueta para trás.



* Insira delicadamente o disco até que ele se fixe numa posição (você deve perceber que existe uma "mola" empurrando o disco para fora e que, quando este atinge sua posição correta, a pressão da mola deixa de existir).
* Feche a porta do drive.

Quando você quiser remover o disco, pressione o botão de liberação logo abaixo da porta do drive (o disco deve então saltar para fora do drive).



ALGUNS CUIDADOS ESPECIAIS

* Quando terminar de usar os discos, deixe-os sempre em seus envelopes.
* Não desligue ou ligue o sistema com um disco no drive.
* Não retire o disco do drive com a lâmpada-piloto acesa.
* Mantenha os discos longe de campos magnéticos como os gerados por transformadores, motores, ímãs, TVs, rádios etc.
* Manuseie os discos com muito cuidado, não tocando nas superfícies expostas em hipótese alguma, mesmo que seja apenas para limpá-las.
* Evite a exposição do disco à fumaça de cigarro, poeira ou qualquer outro tipo de poluição.
* Usar apenas canetas com pontas porosas (macias) para escrever sobre a etiqueta dos discos. De preferência, sempre que possível, escrever antes de colar a etiqueta sobre a jaqueta.
* Guardar os discos na posição vertical.

CAPÍTULO 2

o que é um disco

Um disco magnético é um dispositivo de gravação parecido com uma fita cassete, no que diz respeito ao sistema de gravação. Mas as semelhanças param por aí. Devido ao seu formato, os discos permitem uma utilização muito mais sofisticada e eficiente do que as fitas, principalmente no que se refere à localização das informações.
O disco pode armazenar 161.280 bytes, que equivalem a 1.290.240 de bits ou sinais magnéticos gravados em sua superfície.
Com mais de um milhão de bits em cada disco, você deve estar imaginando como seu computador faz para organizá-los. Ele resolve isto construindo um sistema de arquivo no disco. Primeiro, ele cria os armários de arquivo dividindo seu disco em "trilhas". Em seguida, ele coloca gavetas nos armários dividindo cada trilha em "setores".
Aí, então... nós não terminamos ainda... cada setor é dividido em bytes e cada byte em bits.

Nota: Para ser preciso, há 35 trilhas em um disco, 18 setores em cada trilha, 256 bytes em cada setor e 8 bits em cada byte.

Depois de criar este sistema de arquivo, o computador coloca no disco um índice remissivo (diretório) que indica tudo que está armazenado nele — programas e dados, tais como: lista de endereços, cartas. Ele usa o diretório para encontrar as trilhas e setores onde essas informações foram armazenadas. Este método de arquivamento é, com certeza, o responsável pela grande potência do sistema de discos. O acesso às informações armazenadas é rápido e direto.
A introdução desse sistema de arquivamento em seu disco é chamado "formatação". A última coisa que nós dissemos para você fazer no Capítulo 1 foi inserir um disco "não formatado". Será necessário formatá-lo em trilhas e setores antes de usá-lo.

FORMATANDO UM DISCO

Como você formata um disco? Bem... que tal deixar essa tarefa a cargo do seu computador, hein? Se você foi direto às instruções do último capítulo, já está com o seu sistema ligado e com um disco "não formatado" no drive. E a porta do drive? Está fechada?
Agora digite qualquer letra e pressione a tecla ENTER de modo que:

OK

seja a última linha na sua tela. (OK significa "OK, estou pronto para fazer alguma coisa".) Agora, digite o que você quiser que ele faça. Digite:

DSKINI0

e pressione a tecla ENTER. É possível que apareça na tela a mensagem erro ERROR?SN. Não se preocupe com isso. Foi apenas um "deslize" na hora de digitar o comando. Tente novamente. Essa mensagem aparecerá sempre que alguma coisa sair errado. Deste modo, você pode corrigir o erro imediatamente. Existem, é claro, outras mensagens de erro. Elas estão relacionadas no Apêndice G, juntamente com os seus significados. Depois de digitar DSKINI0 ENTER, você escutará alguns ruídos no seu drive de disco e uma luz vermelha acenderá. Após aproximadamente 40 segundos, seu computador imprime OK. Ele terminou de formatar o disco. Lembre-se de que, sem essa formatação, o armazenamento de informações em disco não é possível. Se você receber um novo, não formatado, você precisará formatá-lo antes de usar.
Mais tarde você pode não se lembrar se um disco foi formatado. Um modo rápido de descobrir isso consiste na verificação do diretório. (Veja "Verificando o Diretório Principal" no fim deste capítulo.) Se você obtiver uma "mensagem de erro", o disco não estará formatado.

Nota: Não há nenhum mal em reformatar um disco. Esse procedimento é usado quando desejamos apagar seu conteúdo.

Se você tiver mais de um drive de disco, poderá formatar um disco em outro drive. Para isso, substitua no comando o número do drive apropriado. Por exemplo, DSKINI1 formata o disco no drive 1.

COLOCANDO UM ARQUIVO EM UM DISCO

Um arquivo de disco pode conter qualquer espécie de informação — um programa, uma lista de endereços, um texto ou algumas anotações. Seu primeiro arquivo conterá um programa BASIC, já que este é o mais simples de se armazenar. Se você não sabe programar em BASIC, simplesmente digite cada linha do programa exatamente como ela é mostrada a seguir. Pressione a tecla ENTER depois de digitar cada linha. Digite:

```
10 PRINT "GUARDE-ME EM UM ARQUIVO DE DISCO" ENTER
20 PRINT "E VOCE NUNCA ME PERDERA" ENTER
```

Acabou? Agora que você digitou o programa na memória do computador, pode colocá-lo em um disco. Para isto, chamaremos esse programa de arquivo e daremos um nome a ele. Por exemplo, "DISCO/PRO" (todos os arquivos têm um nome). Para armazená-lo, digite:

```
SAVE "DISCO/PRO" ENTER
```

Uma vez pressionada a tecla ENTER, seu drive de disco emitirá alguns sons e a luz vermelha acenderá. Seu computador está:
- procurando um lugar no disco para armazenar "DISCO/PRO";
- dizendo ao diretório onde "DISCO/PRO" será armazenado;
- e armazenando "DISCO/PRO" em seu disco.

Nota: Você está ampliando seu sistema a fita? Note a diferença: SAVE armazena um programa no disco; CSAVE armazena-o na fita.

Neste ponto, nós prevenimo-lo de algumas coisas. Não remova seu disco enquanto a luz vermelha estiver acesa. Isso confunde o computador. Ele pode distorcer o conteúdo, não apenas do arquivo que você está armazenando, mas de outros já armazenados no seu disco.
Quando seu computador termina de armazenar "DISCO/PRO", ele imprime a mensagem OK na tela.

Nota: O computador armazena "DISCO/PRO" do mesmo modo que ele armazena todas as coisas — em um código de cargas magnéticas.

MEMÓRIA × ARMAZENAMENTO EM DISCO

Para aqueles iniciantes em computação, gostaríamos de falar um pouco sobre a memória do computador. Caso já conheça o assunto, passe para o próximo item — "Carregando um Arquivo do Disco".
Quando você digitar uma linha de programa em BASIC e pressionar ENTER, o computador a colocará automaticamente na sua memória. Você poderá manipulá-la como quiser. Por exemplo, digite:

```
RUN ENTER
```

Seu computador imprime:

```
GUARDE-ME EM UM ARQUIVO DE DISCO
E VOCE NUNCA ME PERDERA
```

Para listar o programa, digite:

LIST ENTER

A memória é o lugar onde o computador guarda tudo que você conta a ele. Toda vez que você coloca informações na memória do computador, ele pode imprimi-las, rearranjá-las, combiná-las, ou fazer qualquer outra coisa que você deseja.
Se, mais tarde, for necessária a introdução de outro tipo de informação na memória (como, por exemplo, uma mala direta), você precisará escrever ou comprar um programa especialmente estruturado para esse fim. Esse programa aplicativo fará com que o computador coloque na memória as informações digitadas.
Outra característica importante da memória se refere à sua volatilidade. Uma vez apagado o seu conteúdo, não há jeito de recuperá-lo. Uma cópia permanente das informações que você digitou somente será possível por intermédio do seu armazenamento em disco (ou fita).

CARREGANDO UM ARQUIVO DO DISCO

Digite NEW ENTER para limpar tudo na memória do seu computador. Para ter certeza de que tudo foi apagado, digite um ou ambos os comandos:

RUN ENTER
LIST ENTER

Embora NEW limpe o programa da memória, "DISCO/PRO" já está a salvo no seu disco. O "DISCO/PRO" poderá voltar para a memória toda vez que você quiser. Basta "carregá-lo" do disco.
Para fazer isto, digite LOAD "DISCO/PRO" ENTER.
Novamente, você escutará um ruído em seu drive. O computador estará:
- lendo o diretório para encontrar onde "DISCO/PRO" está armazenado no disco;
- indo para esse local do disco e lendo o conteúdo de "DISCO/PRO";
- e colocando "DISCO/PRO" em sua memória.

Você pode digitar agora um ou ambos comandos para verificar se "DISCO/PRO" está na memória:

LIST ENTER
RUN ENTER

MAIS SOBRE ARMAZENAMENTO EM DISCO × MEMÓRIA

Se você está um pouco confuso sobre o que está na memória e o que está no disco, tente este exercício. Você já carregou um programa chamado "DISCO/PRO" na memória, certo? Altere-o digitando:

20 PRINT "COM ESTA MUDANCA" ENTER

Liste o programa novamente para ver se o computador registrou a alteração na linha 20:

```
10 PRINT "GUARDE-ME EM UM ARQUIVO DE DISCO"
20 PRINT "COM ESTA MUDANCA"
```

Armazene-o em um arquivo diferente digitando SAVE "MUDAR" ENTER...
Escutou o barulho no seu drive de disco? Você tem agora dois arquivos: "DISCO/PRO" e "MUDAR". O que você acha que cada um contém? Tente carregar e listar ambos.

Nota: Você não precisa digitar NEW ENTER antes de carregar um programa novo na memória. O computador limpa automaticamente tudo que tem na memória antes de carregar o novo programa.

"MUDAR" contém o novo programa:

```
10 PRINT "GUARDE-ME EM UM ARQUIVO DE DISCO"
20 PRINT "COM ESTA MUDANCA"
```

Entretanto, "DISCO/PRO" contém o programa antigo:

```
10 PRINT "GUARDE-ME EM UM ARQUIVO DE DISCO"
20 PRINT "E VOCE NUNCA ME PERDERA"
```

O único modo de mudar um arquivo de disco consiste em... bem, você sabe. O que você pode fazer para que o arquivo "DISCO/PRO" contenha:

10 PRINT "ARQUIVO TROCADO"

Resposta:
Digite:

NEW ENTER
10 PRINT "ARQUIVO TROCADO" ENTER
SAVE "DISCO/PRO" ENTER

NOMES DE ARQUIVOS

Você já usou o nome de arquivo:

"DISCO/PRO"

E, se você fez nosso exercício de memória × armazenamento em disco, utilizou também o nome:

"MUDAR"

Demos ao nome "DISCO" uma extensão — "PRO". Você deve dar a tudo que você armazena um nome. A extensão é adicional e opcional. Quais nomes você pode dar a seus arquivos? Qualquer um, desde que siga estas regras:
1. O nome não pode ter mais que oito caracteres.
2. Caso ele possua uma extensão, ela não pode ter mais que 3 caracteres.
3. Uma barra (/) ou um ponto (.) deve separar o nome e a extensão.
Combinado? OK.

Nota: Você pode usar qualquer caractere no nome de arquivo, exceto dois pontos (:) ou um zero (0). Somente a barra (/) ou o ponto (.) poderão separar o nome da extensão.

NOMES DE ARQUIVOS QUANDO VOCÊ TEM MAIS QUE UM DRIVE

Se você tem mais que um drive de disco, poderá adicionar o número do drive a seu nome de arquivo. (Lembre-se de que você numerou todos os seus drives no Capítulo 1.) Por exemplo:

LOAD "DISCO/PRO:1"

carrega "DISCO/PRO" do disco no drive número 1.

SAVE "MUDAR:1"

armazena "MUDAR" no disco do drive número 1. Se você não inclui um número de drive, o computador supõe que ele deve usar o drive número 0.

VERIFICANDO O DIRETÓRIO PRINCIPAL

Como nós dissemos anteriormente, o computador usa o diretório principal para procurar o que está no disco. Se ele pode usá-lo, você também pode. Digite DIR ENTER.
O computador exibe informações sobre todos os arquivos armazenados em seu disco. Se você armazenou somente os arquivos "DISCO/PRO" e "MUDAR", aparecerá na tela:

```
DISCO PRO    0   B   1
MUDAR BAS    0   B   1
```

A primeira e a segunda colunas contêm o nome de arquivo. A primeira é o nome e a segunda a extensão. Note que o próprio computador se encarrega de designar uma extensão para o arquivo quando ela é omitida. No caso do arquivo MUDAR, por exemplo, ele atribuiu a extensão BAS. O computador prefere que todos os nomes de arquivos tenham uma extensão. Se você não der ao arquivo uma extensão ao armazená-lo, o computador atribuirá automaticamente uma dessas extensões:

"BAS" se for um programa BASIC;
"DAT" se for um dado (como nomes, números etc.);
"BIN" se for um programa em linguagem de máquina.

Nota: Um programa em linguagem de máquina é um programa altamente técnico que fala diretamente com o computador.

As próximas três colunas contêm informações que deverão despertar interesse somente em técnicos. Se for o seu caso, aqui estão os significados desses códigos.
A terceira coluna contém o tipo de arquivo:

0 programa BASIC
1 dados criados por um programa BASIC
2 dados criados por um programa em linguagem de máquina
3 um programa fonte criado por um editor/assembler

Nota: Um editor/assembler é um programa que você pode comprar para ajudá-lo a criar um programa em linguagem de máquina.

A quarta coluna compreende o formato em que o arquivo é armazenado:
A ASCII
B Binário
Explicaremos mais detalhadamente o significado dessa coluna no Capítulo 10.
A quinta coluna mostra quantos blocos cada arquivo consome. "SIMPLE/PRO" e "CHANGE/BAS" consomem um bloco cada. (O computador usa blocos para alocar o espaço de arquivo em disco. Um disco contém 68 destes blocos.) Se você tiver discos inseridos e formatados em outros drives, poderá também verificar seus diretórios. Por exemplo, DIR 1 ENTER exibe o diretório do disco no drive número 1.
Impressionado? Você ficará bem mais entusiasmado quando ver a rapidez em que se pode preservar (SAVE) e carregar (LOAD) longos programas. Mas antes disso leia o próximo capítulo. Ele lhe ajudará a assegurar um conhecimento sólido sobre seu Sistema de Disco.

Nota: O pressionamento das teclas SHIFT e @ simultaneamente congela a exibição do diretório. Pressione BREAK para liberá-la.

EXERCÍCIO DO CAPÍTULO 2

1. Por que você não pode armazenar dados em um disco não formatado?
2. O que é o diretório do disco?
3. O que é um arquivo de disco?
4. Qual é a diferença entre o que está na memória e o que está no disco?
5. Como você muda o conteúdo de um arquivo em disco?
Você gosta de testes? As respostas estão no Apêndice B.

CAPÍTULO 3

cuidados com o disco

A natureza do material usado para fabricar os discos e a vulnerabilidade das cargas magnéticas o tornam um dispositivo muito delicado. Qualquer pequena partícula como poeira ou fumaça de cigarro prejudica seu conteúdo. Um risco pode arruiná-lo. É por isso que nós sugerimos manter o disco sempre guardado em seu envelope quando ele não estiver em uso — de preferência em um ambiente livre de poeira — e somente usar uma caneta de ponta porosa quando for marcar a etiqueta.
Para ajudar a proteger o disco, ele é revestido por uma jaqueta plástica. Entretanto, como você pode ver, nós não pudemos cobrir o disco inteiro. A secção média e duas outras pequenas áreas são os acessos ao mecanismo de leitura e gravação do computador. Cuidado para não tocar a área exposta, nem sujá-la. Ela é muito vulnerável. Não coloque o disco próximo a outros dispositivos magnéticos, tal como sua televisão. Isto pode alterar completamente os códigos gravados, e portanto causar danos permanentes em suas informações. O calor e a luz do sol podem produzir o mesmo efeito. O mesmo acontece se ligarmos ou desligarmos o computador com o disco no drive.
Outra coisa... Se você está no meio da execução de um programa e precisa trocar o disco, recomendamos a digitação deste comando:

UNLOAD ENTER

antes de trocá-lo. Deste modo o computador pode colocar sua informação de fechamento (CLOSE) no disco apropriado. Se você não digitar este comando, o computador pode colocar esta informação no disco errado e estragar o conteúdo de ambos.

Nota para programadores em BASIC: Todos os arquivos abertos devem ser fechados antes de trocar o disco. UNLOAD fecha todos os arquivos abertos.

CÓPIAS

O BACKUP possibilita a "duplicação" de qualquer um de seus discos, copiando o seu conteúdo em outro disco.
Sugerimos a você fazer um BACKUP de qualquer disco que contenha programas ou dados importantes. Deste modo, você não se preocupará em perder informações.
Outro ponto a considerar se refere ao desgaste do disco pelo próprio uso. Uma cópia do disco velho será sempre bem-vinda. Assim quando o computador tiver problemas na leitura ou gravação do disco, você poderá usar a cópia nova.
Você quer fazer uma cópia? Então prepare dois discos:

1. Seu disco "fonte" — Isto é, o disco que você quer duplicar. Use qualquer disco que tenha arquivos armazenados. Se você é um principiante, use o disco com que trabalhou no Capítulo 2.

2. Seu disco "destino" — É o disco que você quer que seja sua cópia. Use um disco virgem ou qualquer outro que contenha arquivos dispensáveis.

Nota: Todas as informações antigas do seu disco "destino" serão apagadas e substituídas pelos dados do seu disco "fonte".

Se seu disco "destino" é virgem, você deve primeiro formatá-lo. Você se lembra como? Insira-o em seu drive de disco, feche a porta e digite DSKINI0 ENTER.
Agora faça a cópia (BACKUP). O procedimento a seguir depende do número de drives do seu sistema.

BACKUP COM UM DRIVE DE DISCO

Se houver apenas um drive de disco, você levará aproximadamente 5 minutos para fazer um BACKUP. Insira seu disco "fonte" em seu drive de disco e feche a PORTA DO DRIVE. Digite DIR ENTER para ver quais arquivos serão copiados.
Agora, inicie os procedimentos de BACKUP. Digite:

BACKUP 0 ENTER

Após a leitura de uma parte de seu disco "fonte", o computador imprimirá:

INSERIR DISKETE DESTINO E PRESSIONAR ENTER

Tire o disco "fonte", insira o disco "destino" e feche a PORTA DO DRIVE. Depois, pressione ENTER. O computador grava algumas informações no disco "destino" e imprime:

INSERIR DISKETE FONTE E PRESSIONAR ENTER

O computador o orientará a trocar os discos até que tenha copiado tudo do disco "fonte". Durante esse processo, certifique-se sempre de que você inseriu o disco certo de modo correto. Quando você terminar o computador imprimirá a mensagem OK na tela.
Insira seu disco "destino" no drive e digite DIR ENTER, para verificar seu bom funcionamento.

BACKUP COM MAIS DE UM DRIVE DE DISCO

Num sistema com mais de um drive de disco, a cópia é muito mais fácil. Geralmente, ela leva apenas 2 minutos.
Insira seu disco "fonte" no drive 0, e seu disco "destino" no drive 1. (O Capítulo 1 mostra como rotular seus drives.) Depois, digite:

BACKUP 0 TO 1 ENTER

Você escutará algum ruído quando o computador copiar o conteúdo do disco do drive 0 no disco do drive 1. Ele imprime a mensagem OK quando termina o processo. Certifique-se de que seu BACKUP foi aceito digitando DIR1 ENTER.
Você pode também usar drives diferentes. Por exemplo:

BACKUP 1 TO 0 ENTER

copia o conteúdo do disco do drive 1 no disco no drive 0.

PROBLEMAS DURANTE O BACKUP

Durante a cópia, poderão aparecer mensagens de erro devido à inserção incorreta do disco ou a problemas com o próprio disco.
No fim deste capítulo, veremos as várias mensagens de erro que lhe ajudarão a determinar o problema. Se você tiver um disco ruim, terá que tentar um BACKUP com outro.
Depois de determinar o problema, pressione as duas teclas RESET para sair de BACKUP, e recomece o procedimento de BACKUP novamente.

Nota: As teclas RESET estão à direita do teclado.

PROTEÇÃO CONTRA GRAVAÇÃO

Seu disco necessita de algum tipo de proteção contra gravações acidentais. Imagine, por exemplo, que você possua um disco com um programa muito importante e de uso diário. Você sempre o carrega na memória, mas não tem intenção de gravar nada nesse disco.
Se você observar mais atentamente um disquete, descobrirá que nele existe um entalhe de proteção contra gravação. Quando devidamente selado, o computador só consegue ler a informação que ele contém. Esses selos geralmente acompanham os discos virgens, mas qualquer outro semelhante pode ser usado.

SALVANDO-O

Nada é eterno, mas antes de jogar fora um disco antigo, veja se não dá para recuperá-lo. Você pode fazer isso formatando-o de novo como se fosse um disco vazio. Embora isto salve o disco, o seu conteúdo estará perdido. De qualquer modo, essa medida evita gastos na compra de novos discos.

Um erro IO indicará provavelmente que o disco alcançou o seu limite de uso. (Veja "Mensagens de Erro" no fim deste capítulo.) Se você tem um "apagador", pode tentar "apagar" o disco e reformatá-lo. Caso contrário retire-o e use um outro.

Nota: Se você tiver mais de um drive, poderá copiar (COPY) alguns dos arquivos de um disco defeituoso em outro que esteja em melhores condições. Discutiremos COPY no próximo capítulo.

VERIFICAÇÃO

O computador "grava" dados em alta velocidade no seu disco. Na maioria das vezes, isso ocorre sem problemas. Porém, às vezes, você necessita se certificar da exatidão dos dados gravados.
O comando VERIFY existe para esse fim. Digite:

VERIFY ON ENTER

O computador indica se há algum defeito no disco em que ele está gravando. O único problema se refere ao tempo de gravação que é duplicado. Este comando VERIFY permanece ligado até sua desativação. Para fazer isto, digite:

VERIFY OFF ENTER

PROBLEMAS

Seu computador sabe que ninguém é perfeito. Quando você comete um erro, ele o indica imediatamente. Isto já deve ter acontecido com você. Se não, digite DIIR ENTER, e verá aparecer um "?SN ERRO" na tela. SN significa erro de "Sintaxe". É o jeito de o computador contar a você que "DIIR" não faz sentido para ele. A palavra não está no seu vocabulário. Um erro SN geralmente implica um erro ortográfico.
Aqui estão algumas mensagens de erro que você provavelmente obterá com seu sistema de disco:

AE — Você está tentando dar um novo nome (RENAME) a um arquivo (discutiremos no próximo capítulo) e esse nome já existe.

DF — O disco que você está tentando armazenar em seu arquivo já está cheio. Use outro disco.

DN — Você está usando um número de drive maior que 3. Esse erro ocorrerá também se você não especificar o número de drive quando usar DSKINI ou BACKUP. Se você tem apenas um drive, especifique o drive 0 com estes dois comandos (DSKINI0 ou BACKUP 0).

FN — Você usou um formato ilegal para dar nome ao seu arquivo. O último capítulo explica quais nomes de arquivo são aceitos pelo computador.

FS — Há alguma coisa errada com seu arquivo de disco. Veja IO para saber o que fazer.

/0 — Tecnicamente, isto significa que você pediu ao computador para dividir um número por zero, o que é impossível. Entretanto, você pode obter este erro quando você não coloca um nome de arquivo entre aspas.

IO — O computador está tendo problemas com a entrada e saída de informações para o disco.
(1) Certifique-se de que o disco foi inserido corretamente no drive apropriado e se a porta do drive está fechada.
(2) Se o erro persiste, deve haver algo errado com seu disco. Primeiramente, tente reinserir o disco. Depois, usar um outro ou reformatá-lo. (Lembre-se que isso limpa seu conteúdo.)
(3) Se o erro ainda persiste você provavelmente tem um problema com o sistema do computador.

NE — O computador não pode encontrar o arquivo de disco que você quer. Verifique o diretório do disco para ver se ele está lá. Se você tem mais de um drive de disco, talvez você não tenha incluído o número de drive apropriado no nome do arquivo. Se você está usando COPY, KILL, ou RENAME (serão discutidos no próximo capítulo), você deve ter-se esquecido da extensão.

TM — Tecnicamente, isto é causado por um programa que mistura "strings" com "números". Entretanto, você pode obter este erro se não colocar um nome de arquivo entre aspas.

VF — Esse erro só acontece quando o comando VERIFY estiver ativado ao mesmo tempo em que se grava alguma coisa num disco. O computador está lhe informando que há um defeito nas informações gravadas. Veja IO para saber o que fazer.

WP — Você está tentando armazenar informação em um disco que está protegido contra gravação. Tire o selo de proteção ou use um disco diferente. Se seu disco não está protegido contra gravação, então há um problema de entrada/saída. Ver IO para saber o que fazer.

Todos os outros erros que você pode obter são erros no programa que você está usando. Se você não escreveu o programa e obteve um desses erros, contate o pessoal responsável. Se você o escreveu, verifique o Apêndice H, onde encontrará uma explicação de todas as mensagens de erro.

Os cuidados com seus discos podem parecer complicados à primeira vista. Você passou a maior parte de sua vida protegendo seus papéis e agora está lidando com um meio diferente.

Depois de algum tempo, contudo, proteger seu disco de poeira e dispositivos magnéticos parecerá tão natural quanto proteger seus papéis contra rajadas de vento. E depois que se acostumar a cuidar do seu disco, nunca mais vai querer voltar ao lápis e papel.

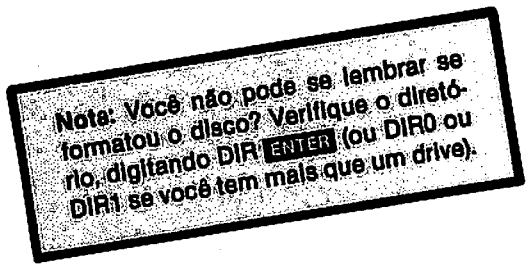
EXERCÍCIOS DO CAPÍTULO 3

1. Por que você não pode ligar ou desligar seu computador enquanto o disco estiver no seu drive?
2. Qual o tipo de caneta que você pode usar para escrever na etiqueta do disco?
3. O que é mensagem de erro?
4. O que significa proteção contra gravação? Como fazer isso?
5. Como você faz uma cópia (BACKUP) de um disco?

CAPÍTULO 4

você é o chefe

Graças ao seu sistema de arquivo em disco, você está apto a fazer uma porção de coisas interessantes com o seu computador. Por exemplo, você pode dar um outro nome a seu arquivo. Coloque o disco formatado no drive de disco.



Digite isto para colocar um arquivo em seu disco:

10 PRINT "ESTE E UM ARQUIVO" ENTER
SAVE "ORIGINAL/NOM" ENTER

Verifique o diretório para ver se o arquivo de programa está armazenado em seu disco sob o nome "ORIGINAL/NOM"... Agora, dê-lhe outro nome. Digite:

RENAME "ORIGINAL/NOM" TO "NOVO/NOM" ENTER

Veja se o drive de disco está funcionando. Verifique seu diretório novamente. Carregue e liste o "NOVO/NOM". O arquivo de programa simplesmente tem outro nome. O resto é idêntico.
RENAME é fácil de usar, mas há outra coisa de que você precisa se lembrar. Preserve um arquivo sem uma extensão e, então, tente dar-lhe outro nome. Digite:

10 PRINT "ARQUIVO NUMERO DOIS" ENTER
SAVE "ARQA" ENTER
RENAME "ARQA" TO "ARQB" ENTER

O computador indica um erro NE. Isto significa que o computador não pode encontrar o arquivo.
Ao dar um outro nome a um arquivo, você deve digitar o nome completo do arquivo. Só assim o computador pode achá-lo. Isto inclui a extensão. Como discutimos no Capítulo 2, quando você preserva um arquivo, o computador se certifica se ele tem uma extensão. Se você não determina uma, o computador o faz.
Você pode verificar o diretório para descobrir a extensão de "ARQA". Então, substitua o nome antigo pelo novo. Digite:

RENAME "ARQA/BAS" TO "ARQB/BAS" ENTER

Certifique-se de que seu novo nome de arquivo tenha uma extensão. Em outras palavras, não digite:

RENAME "ARQA/BAS" TO "ARQB" ENTER

O computador dá ao arquivo o novo nome, porém "ARQB" fica sem extensão. Isto causará um problema quando você tentar carregar "ARQB", já que todos os arquivos que você carrega devem ter uma extensão. Isto pode parecer conflitante com o que dissemos anteriormente. Você podia preservar "ARQA" sem determinar uma extensão porque o computador fazia isso automaticamente. RENAME funciona de forma diferente. O programa não atribui automaticamente uma extensão para o novo nome do programa.

Nota: Há um modo de carregar "ARQB" sem uma extensão. Isto é feito indicando a ausência de extensão ao digitar LOAD "ARQB/" ENTER. Isto é desaconselhável. Por isso, aconselhamos a atribuição de uma extensão em todos os casos.

VÁRIOS DRIVES DE DISCO

Você pode alterar o nome de um arquivo em outro drive de disco, simplesmente digitando o número do drive apropriado. Insira um disco formatado no drive 1. Armazene um arquivo nele:

10 PRINT "CONTABILIDADE" ENTER
SAVE "VELCON/DAT:1" ENTER

e altere seu nome digitando:

RENAME "VELCON/DAT:1" TO "NOVCON/DAT:1" ENTER

Nota: Se você deseja alterar o nome do seu arquivo e, ao mesmo tempo, copiá-lo em outro drive, o RENAME não pode lhe ajudar. Use COPY.

FALTA MUITO?

Mais cedo ou mais tarde, você vai querer saber qual o espaço útil de armazenamento em seu disco. Digite:

PRINT FREE (0) ENTER

O computador imprime o número de blocos livres restantes no seu disco. São 68 blocos ao todo. Se o computador indica a existência de um único bloco livre (FREE), o melhor que você tem a fazer é o seguinte: comece

a usar outro disco ou a eliminar (KILL) alguns de seus arquivos de disco. Por exemplo, se você colocou "MUDAR" em seu disco no Capítulo 2, digite:

KILL "MUDAR/BAS" ENTER

Verifique seu diretório e o espaço livre (FREE) restante no seu disco. "MUDAR/BAS" não está mais no seu disco. O espaço por ele ocupado está agora livre para novos arquivos.
Note que tivemos de incluir a extensão de MUDAR ("BAS") a fim de eliminá-lo. Essa é uma precaução extra do computador. Ele não quer eliminar um arquivo que você talvez não queria destruir.

Nota: Mais dados técnicos? Na realidade, os dados permanecem no disco mesmo depois da eliminação de um arquivo. Entretanto, o computador não saberá que eles estão lá, porque KILL elimina todas as referências a eles no diretório de disco. Conseqüentemente, você não poderá mais ter acesso aos dados, e o computador estará apto a gravar sobre ele um novo arquivo.

VÁRIOS DRIVES DE DISCO

Você pode usar FREE e KILL em outros drives de disco, e do mesmo modo RENAME, digitando o número do drive. Exemplo:

PRINT FREE (1) ENTER

informa o espaço livre (FREE) no disco do drive 1.

KILL "NOVCON/DAT:1" ENTER

anula "NOVCON/DAT" do disco no drive 1.

COMANDOS ESPECIAIS PARA SISTEMAS DE VÁRIOS DRIVES

No restante deste capítulo, falaremos sobre dois comandos destinados a sistemas de vários drives. Se você não tiver um, passe para a "Verificação do Capítulo". O primeiro comando copia um arquivo de disco. Você pode, neste ponto, ter um arquivo de programa armazenado no disco no drive 0 chamado "NOVO/NOM". Faça uma cópia dele. Digite:

COPY "NOVO/NOM:0" TO "NOVO/NOM:1" ENTER

Se você quiser, você pode alterar o nome do arquivo ao copiá-lo. Por exemplo, COPY "NOVO/NOM:1" TO "OUTRO/NOM:0" ENTER copia o "NOVO/NOM" do disco no drive 1 no arquivo "OUTRO/NOM" do disco no drive 0.
O segundo comando altera o número do drive com que o computador trabalha caso ele não seja especificado. Até agora, o único drive utilizado tem sido 0. Por exemplo, quando você digita SAVE "QULCOISA/EX" ENTER, o computador supõe que você deseja usar o drive 0. E ele preservará esse programa no disco do drive 0.
Porém, se você digitar:

DRIVE 1 ENTER

o computador pensará que você quer usar o DRIVE 1.
Ele armazenará "QULCOISA/EX" no disco do drive 1. Você precisará agora digitar SAVE "QULCOISA/EX:0" ENTER para preservá-lo (SAVE) no drive 0.

EXERCÍCIOS DO CAPÍTULO

1. *Como você altera o nome de um arquivo? Por que você tem que especificar a extensão do arquivo?*
2. *O que você pode fazer quando percebe que o espaço útil de armazenamento está quase no fim?*
3. *Qual drive o computador usará se você tiver mais de um drive de disco e não especificar o seu número? Como você pode mudar isso?*

CAPÍTULO 5

arquivo de acesso seqüencial

Neste capítulo, mostraremos como gravar um programa que armazena dados em um arquivo de acesso seqüencial em disco. É o arquivo mais simples de se criar e é muito similar ao "arquivo" em fita. No próximo capítulo, introduziremos o "acesso direto", uma alternativa de arquivo em disco. Ao descrevermos todo o mecanismo de armazenamento de informação em disco utilizaremos freqüentemente as palavras **arquivo de disco** e **diretório de disco**. Já discutimos estes conceitos no Capítulo 2, mas faremos um resumo agora.

Tudo que armazenamos em um disco deve estar em um **arquivo de disco** e ter um nome de arquivo. Seu computador indica a localização do arquivo de disco por intermédio do **diretório de disco**. Assim, tudo que for armazenado em um disco deve ter um nome. Com apenas um comando (DIR) você fica sabendo todo o conteúdo do seu disco.

GRAVANDO UM ARQUIVO DE DISCO

Suponha que você deseja gravar seus cheques em um disco:

CHEQUES

"DR. EDSON"
RESTAURANTE
SINDICATO
ESCRITORIO

Iniciaremos com um programa pequeno e simples que grava o primeiro cheque ("DR. EDSON") no disco. Insira um disco formatado no seu drive. (Se você tiver mais de um drive de disco, use o drive 0.)

Nota: O Capítulo 2 ensina como formatar um disco. (Digite DIR ENTER se você não se lembra se o disco foi formatado.) O Capítulo 1 explica os números do drive.

Depois digite:

```
10 OPEN "O", # 1, "CHEQUES/DAT"
20 WRITE #1, "DR. EDSON"
30 CLOSE #1
```

Rode o programa. Você escutará o barulho do motor do drive de disco e verá a luz vermelha se acender. O computador está efetuando várias tarefas. Primeiro, ele libera (OPEN) a comunicação com o disco para que você possa enviar seus cheques para ele. Depois, encontra uma área desocupada para armazenar os cheques, e anota o início dessa área do disco diretório.

Tudo isto acontece na linha 10. Note o significado de #1, "O" e "CHEQUES/DAT":

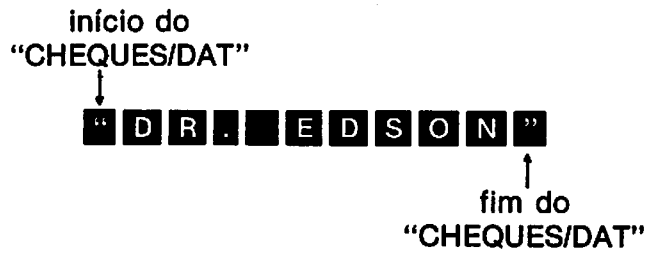
1. #1 é uma área especial na memória chamada buffer #1. Ele se comunica com o drive de disco. A linha 10 abre (OPEN) esse buffer. (Se você já trabalhou com fita, deve se lembrar que o buffer # – 1 se comunica com o gravador de fita.)
2. "O" é a letra "O", não um zero. Ela significa saída, e diz ao computador que o buffer #1 enviará dados ao disco.
3. "CHEQUES/DAT" é o nome do arquivo de disco. O diretório do disco usa este nome para indicar o seu início e fim.

Na linha 20, o computador envia as palavras "DR. EDSON" ao buffer #1 para depois gravá-las (WRITE) no disco.
Então, na linha 30 o computador inibe (CLOSE) a comunicação com o buffer #1. Fazendo isto, ele:

- envia todos os dados restantes no buffer #1 ao arquivo de disco;
- anota no diretório de disco onde o "CHEQUES/DAT" termina.

Nota: Um buffer armazena temporariamente dados, assim o computador pode introduzir e retirar dados do disco em blocos de 249 caracteres (bytes). Como o buffer #1 tem apenas 9 caracteres ("DR. EDSON"), eles não serão enviados ao disco sem que se peça para fechar o arquivo.

É muito importante que você iniba (CLOSE) a comunicação com o buffer #1. Por quê? Bem, vamos deixar o buffer #1 aberto (OPEN) e eliminar a linha 30. Rode o programa várias vezes.
Parece que o programa está funcionando do mesmo jeito todas as vezes que você o executa. Isto porque toda vez que você roda (RUN) ou carrega (LOAD) um programa, o computador inibe (CLOSE) automaticamente a comunicação com qualquer buffer que você tenha deixado aberto (OPEN).
Agora, vamos supor que você troque o disco e rode (RUN) ou carregue (LOAD) um programa. O computador inibe automaticamente a comunicação com o buffer #1. Ao fazer isto, ele envia sua informação de inibição ao novo disco (pensando que é o velho). Isto provavelmente alterará os conteúdos dos dois discos.
Agora que nós o alertamos a respeito da importância da linha 30, reescreva-a no programa e rode-o novamente. Isto é o que o programa escreve no seu disco:



Nota: O conteúdo do arquivo "CHEQUES/DAT" são as palavras "DR. EDSON". O diretório de disco anota a localização do início e do fim deste arquivo. Você pode verificar se o computador fez isso, checando o seu diretório. Você se lembra como fazer isto? (Digite DIR ENTER.)

Este programa envia dados para o arquivo de disco, portanto, o chamaremos programa de **saída.**

LENDO UM ARQUIVO DE DISCO

Para fazer com que o computador carregue esses dados na sua memória, você precisa de um programa de **entrada**. Digite NEW ENTER para limpar a memória do computador. Depois, digite e rode este programa de entrada:

```
100 OPEN "I", #1, "CHEQUES/DAT"
110 INPUT #1, A$
120 PRINT A$
130 CLOSE #1
```

Este programa é exatamente o inverso do programa de saída. A linha 100 novamente libera a comunicação com o buffer #1. Desta vez, a comunicação é liberada (OPEN) para "I" — **entrada**. O computador vai ao diretório de disco para descobrir onde ele deve começar a ler o arquivo chamado "CHEQUES/DAT".
Na linha 110, o computador introduz (INPUT) o primeiro dado (A$) do arquivo de disco chamado "CHEQUES/DAT". A linha 120 imprime A$. Finalmente, a linha 130 inibe (CLOSE) a comunicação com o buffer #1. Nesse momento, o computador introduz na memória os dados restantes no buffer.

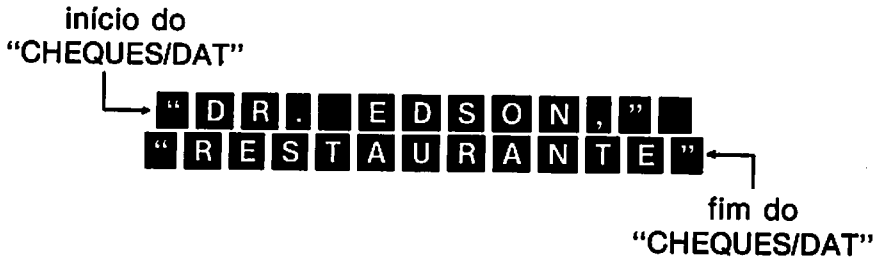
Nota: Você pode comparar um programa de entrada com o comando LOAD. O primeiro lê um dado do arquivo, enquanto que o segundo lê um arquivo de programa.

VERIFICANDO OS DADOS

Até agora usamos um programa de saída e outro de entrada. Vamos combiná-los em um único programa, e acrescentar mais algumas linhas aumentando o seu arquivo de disco.

```
10 OPEN "O", #1, "CHEQUES/DAT"
20 WRITE #1, "DR. EDSON"
25 WRITE #1, "RESTAURANTE"
30 CLOSE "I", #1, "CHEQUES/DAT"
100 OPEN "I", #1, "CHEQUES/DAT"
110 INPUT #1,A$
115 INPUT #1,B$
120 PRINT A$, B$
130 CLOSE #1
```

As linhas de 10 a 30 **colocam** (OUTPUT) dois cheques **no seu arquivo de disco:**



As linhas 100 a 130 **colocam** (INPUT) os cheques **na memória**. Tente introduzir mais que dois cheques. Mude as linhas 115 e 120:

```
115 PRINT A$
120 GOTO 110
```

e rode o programa... O computador imprime:

```
?IE ERRO NA 110
```

Como essa mensagem, o computador lhe diz que foi requisitada a introdução de mais cheques do que os existentes no arquivo. Tecnicamente, o erro IE significa que você tentou introduzir informações depois do Fim de Arquivo.
Este erro implica dificuldades na introdução de dados quando se desconhece sua quantidade. A função EOF resolve esse problema. Digite:

```
105 IF EOF(1) = -1 THEN 130
120 GOTO 105
```

e rode... EOF verifica se o arquivo no buffer #1 (o número entre parênteses) lotou. Se for esse o caso, EOF(1) será igual a −1; se não, ele será igual a zero.
A linha 105 verifica o final do arquivo, antes de introduzir o próximo cheque. Em caso afirmativo, a linha 130 inibe a comunicação com o arquivo.

MAIS ALGUNS DETALHES

PAGO A	CHEQUES QUANTIA	TIPO DE DESPESA
DR. EDSON	50 000,00	DENTISTA
RESTAURANTE	130 000,00	ALIMENTAÇÃO
SINDICATO	29 000,00	CONTRIB.
ESCRITÓRIO	530 000,00	COMERC.

Mude as linhas 25 e 115, e adicione algumas linhas digitando:

```
25 WRITE #1, 50000.00
27 WRITE #1, "DENTISTA"
110 INPUT #1, A$, B, C$
115 PRINT A$, B, C$
```

Liste (LIST) o programa. Ele fica assim:

```
10 OPEN "O", #1, "CHEQUES/DAT"
20 WRITE #1, "DR. EDSON"
25 WRITE #1, 50000.00
27 WRITE #1, "DENTISTA"
30 CLOSE #1
100 OPEN "I", #1, "CHEQUES/DAT"
105 IF EOF(1) = -1 THEN 130
110 INPUT #1, A$, B, C$
115 PRINT A$, B, C$
120 GOTO 105
130 CLOSE #1
```

Agora, rode-o.

UM PROGRAMA COMPACTO

E se você precisar armazenar vários cheques? Desse jeito, você vai ter mais trabalho do que resultados.
Temos aqui um programa compacto que pede a você para introduzir (INPUT) todos os seus dados na memória, armazená-los no disco e colocá-los novamente na memória. Limpe a memória e digite:

```
5 CLS
10 OPEN "O", #1, "CHEQUES/DAT"
20 INPUT "CHEQUE PAGO A :"; A$
30 IF A$ = "" THEN 80
40 INPUT "QUANTIA : $"; B
50 INPUT "DÉSPESA :"; C$
60 WRITE #1, A$, B, C$
70 GOTO 20
80 CLOSE #1
90 CLS
100 PRINT "SEUS CHEQUES ESTAO ARMAZENADOS NO DISCO"
110 INPUT "PRESSIONE <ENTER> PARA LE-LOS"; A$
120 OPEN "I", #1, "CHEQUES/DAT"
130 IF EOF(1) = -1 THEN 170
140 INPUT #1, A$, B, C$
150 PRINT A$; B; C$
160 GOTO 130
170 CLOSE #1
```

Agora rode-o. Introduza qualquer cheque. Quando você quiser desistir, simplesmente pressione ENTER em resposta a CHEQUE PAGO A: ponto de interrogação. Por exemplo:

```
CHEQUE PAGO A: ? VERMELHO ENTER
QUANTIA: $ ? 130000 ENTER
```

DESPESA: ? CARRO ENTER
CHEQUE PAGO A: ? ENTER
SEUS CHEQUES ESTAO ARMAZENADOS NO DISCO
PRESSIONE <ENTER> PARA LE-LOS ? ENTER
AUTO VERMELHO 130000 CARRO

ATUALIZANDO UM ARQUIVO DE ACESSO SEQÜENCIAL

Tudo o que você coloca no disco e tira dele passa através de um ponto na memória chamado buffer.
No manual do seu computador, falamos sobre esse buffers quando discutimos a introdução de dados numa fita. Havia apenas um buffer que se comunicava com a fita — o buffer # – 1.
Com seu sistema de disco, você pode usar até 15 buffers. Isso significa que podem existir até 15 pontos na memória se comunicando com 15 arquivos diferentes de disco ao mesmo tempo.
Usaremos dois buffers para mostrar como se processa a alteração de dados em seu arquivo.

Nota: No Capítulo 10, mostraremos como aproveitar mais estes buffers.

Digite este programa:

```
10 OPEN "O", #1, "CHEQUES/DAT"
20 WRITE #1, "DR. EDSON"
30 WRITE #1, "RESTAURANTE"
40 CLOSE #1
```

Rode-o. Agora, vamos supor que você queira substituir "RESTAURANTE" por "LEILAO". Primeiro, lê-se os dados na memória com um programa de entrada. Depois, limpe a memória com NEW ENTER e digite:

```
10 OPEN "I", #1, "CHEQUES/DAT"
20 IF EOF (1)= –1 THEN 110
30 INPUT #1, A$
40 CLS:PRINT @ 100, "ITEM DE DADOS: ";A$
50 PRINT @ 449, "PRESSIONE <ENTER> SE NAO HOUVER
ALTERACAO";
60 PRINT @ 260, "MUDANCA: ";
70 INPUT X$
80 IF X$ = "" THEN X$ = A$
90 WRITE #1,X$
100 GOTO 20
110 CLOSE #1
```

Rode o programa. Quando pressionar ENTER, o computador executará a linha 90 e imprimirá:

```
?FM ERRO NA 90
```

Liste (LIST) o programa. A linha 10 libera o buffer #1 para introdução de dados, e a linha 90 tenta enviá-los para ele. O computador não envia dados para o buffer que está fechado para entrada (INPUT). É aqui que um buffer adicional se torna útil. Você pode simplesmente abrir outro buffer para saída. Acrescente estas linhas:

```
15 OPEN "O", #2, "NOVO/DAT"
90 WRITE #2, X$
120 CLOSE #2
```

Rode o programa. Mude "RESTAURANTE" para "LEILAO". O programa inteiro fica assim:

```
10 OPEN "I", #1, "CHEQUES/DAT"
15 OPEN "O", #2, "NOVO/DAT"
20 IF EOF (1) = -1 THEN 110
30 INPUT #1, A$
40 CLS:PRINT @ 98, "ITEM DE DADOS: ";A$
50 PRINT @ 449, "PRESSIONE <ENTER> SE NAO HOUVER
ALTERACAO";
60 PRINT @ 260, "MUDANCA: ";
70 INPUT X$
80 IF X$="" THEN X$=A$
90 WRITE #2, X$
100 GOTO 20
110 CLOSE #1
120 CLOSE #2
```

A linha 30 introduz na memória a variável A$ do buffer #1. A linha 70 permite que você introduza (INPUT) X$ para substituir A$. Ao introduzir X$, a linha 90 o envia para o buffer #2, que por sua vez grava-o (WRITE) no "NOVO/DAT". A linha 110 interrompe a comunicação com o buffer #1 e a linha 120, a comunicação com #2.
Agora, você tem dois arquivos. O "CHEQUES/DAT" contém o dado antigo e o "NOVO/DAT", o novo. Acrescente estas linhas ao programa e rode-o:

```
130 KILL "CHEQUES/DAT"
140 RENAME "NOVO/DAT" TO "CHEQUES/DAT"
```

Agora, o arquivo "CHEQUES/DAT" é anulado do disco e o nome do arquivo "NOVO/DAT" passa a ser "CHEQUES/DAT". Para ver o que esse arquivo atualizado contém, preserve (SAVE) esse programa. Caso seja necessário, limpe a memória, digite e rode o programa abaixo:

```
10 OPEN "I", #1, "CHEQUES/DAT"
20 IF EOF(1) = -1 THEN 60
```

```
30 INPUT #1, A$
40 PRINT A$
50 GOTO 20
60 CLOSE #1
```

Entendeu? Tente esse exercício:
Estruture um programa de mala direta, que liste o nome, o endereço e o telefone de seus amigos. O início do programa se encontra logo abaixo. Complete-o e depois compare-o com o nosso Apêndice A.

```
10 OPEN "O", #1, "AMIGOS/DAT"
20 GOSUB 100
30 IF N$ = "" THEN CLOSE #1: END
40 WRITE #1, N$, E$, T$
50 GOTO 20
100 CLS: PRINT "PRESSIONE <ENTER> QUANDO TERMINAR":
PRINT
110 INPUT "NOME: ";N$
120 IF N$ = "" THEN 150
130 INPUT "ENDERECO: "; E$
140 INPUT "TELEFONE: "; T$
150 RETURN
```

FAÇA VOCÊ MESMO #5.1

Escreva um programa em que você pode:
1. Ver os nomes, endereços e telefones dos seus amigos.
2. Alterar endereços ou telefones.
3. Acrescentar novos nomes.
4. Eliminar alguns dos nomes.

Todo esse processo funciona bem em pequena escala. Mas que tal em um arquivo grande? E se você tivesse 500 amigos em seu arquivo e quisesse mudar apenas o endereço do 453º?
O processo ainda seria o mesmo. Você teria que colocar na memória cada um dos 500 amigos constantes num arquivo e, então, enviá-los todos para outro arquivo. Tudo isto só para mudar uma gravação. Deve haver um modo mais fácil!
Sim, existe. Ele é chamado método de acesso direto de programação. Com esse método seus arquivos se tornam mais fáceis e rápidos de atualizar. Porém, em alguns casos, eles ocupam mais espaço no seu disco. A escolha é sua. Nós falaremos sobre o acesso direto no próximo capítulo.

Nota: Aqui, limitamo-nos a pequenos programas exemplo. Há alguns modos de você melhorá-los. Veja "Exemplos de Programa" no Apêndice C.

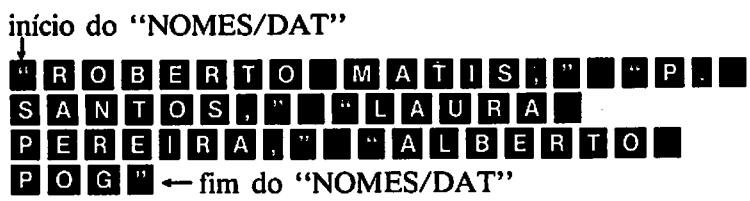
EXERCÍCIOS DO CAPÍTULO

1. *O que é um buffer # 1?*
2. *Quando você deve abrir um arquivo de disco?*
3. *Por que você deve fechá-lo (CLOSE)?*
4. *Qual é a diferença entre um arquivo aberto (OPEN) para entrada e outro aberto para saída?*
5. *Você pode introduzir e enviar dados para o mesmo buffer simultaneamente?*
6. *Você pode introduzir dados de um arquivo aberto (OPEN) para saída ("O" — output)?*

CAPÍTULO 6

arquivo de acesso direto

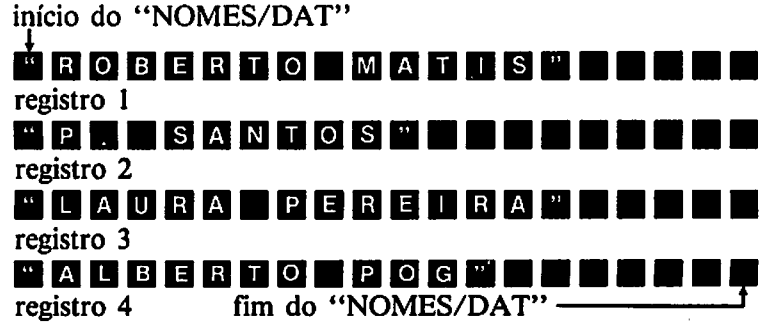
Até agora não nos preocupamos com a organização de seus dados armazenados no disco. Por exemplo, você pode ter colocado isto em um arquivo:



E se você deseja trocar "P. SANTOS" por "ISABEL CIP"? Você não pode pedir ao computador para ir direto a "P. SANTOS". Ele não sabe onde esse item está.
Todos os arquivos que nós criamos foi em "acesso seqüencial". Para encontrar um item num arquivo de acesso seqüencial, o computador precisa ler cada item desde o início até encontrá-lo. Ele não pode ir diretamente ao item. Em resumo, um arquivo de acesso seqüencial não é vantajoso ao seu sistema de arquivo em disco.

USANDO O SISTEMA DE ARQUIVO EM DISCO

No Capítulo 2, falamos como a formatação de seu disco cria este sistema de arquivo. Na nossa analogia, os gabinetes são as "trilhas" do disco, e as gavetas, os "setores". Você pode usar trilhas e setores para encontrar rapidamente qualquer item que você quiser.
Para fazer isto, divida seu arquivo em fichas ou, como é mais conhecido, em "registros". Você pode então gravar um programa que armazena cada registro em um setor e permite que você coloque dados nos registros. A seguir, mostraremos como seu arquivo de disco ficará:



Com cada registro do mesmo tamanho (o tamanho de um setor), o computador pode ir direto a "P. SANTOS". Tudo que ele tem de fazer é contar até o segundo registro.
Esse tipo de arquivo é chamado de "acesso direto". Você pode ter acesso direto a qualquer registro. Um arquivo de acesso direto tem uma restrição. Cada registro tem o tamanho de um setor — 256 bytes. Já que cada byte corresponde a um caractere de dado, cada registro contém 256 caracteres. Isto significa que o nosso exemplo acima não espelha exatamente

o que acontece. Se tivéssemos ilustrado todos os espaços vazios de cada registro, ia ser preciso aproximadamente dez vezes mais espaço na folha. Se você é principiante, todo este espaço vazio provavelmente não o aborrece. Um disco vazio pode conter até 612 registros — cada um com 256 bytes de comprimento. Mais tarde, quando você se sentir mais à vontade com a programação, provavelmente vai querer colocar mais registros no seu arquivo de disco. O Capítulo 8 mostra como diminuir o tamanho dos registros.

COLOCANDO UM REGISTRO NO DISCO

Chega de teoria! Vamos colocar um registro em um arquivo de disco. Já que o arquivo vai ser de acesso direto, nós não teremos que iniciar com o primeiro. Começaremos pelo segundo. Limpe a memória e digite:

```
10 OPEN "D", #1,"NOMES/DAT"
20 WRITE #1, "P.SANTOS"
30 PUT #1,2
34 GET #1,2
36 INPUT #1,A$
38 PRINT A$
40 CLOSE #1
```

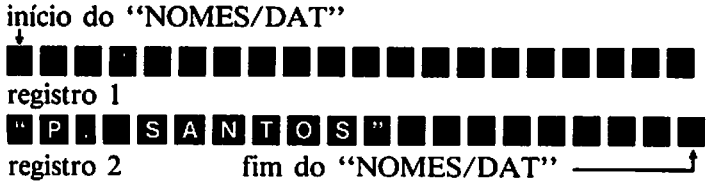
Rode-o... Você escutará o som familiar de seu drive de disco. O computador está gravando "P. SANTOS" no arquivo de disco e, depois, gravando-o novamente na memória. Veja como...

A linha 10 libera (OPEN) o buffer #1 que se comunica com um arquivo de disco chamado "NOMES/DAT". Como dissemos no último capítulo, o buffer #1 é uma das 15 áreas de buffer que podem se comunicar com seu disco.

A comunicação é liberada para "D" (acesso direto). Aqui, você não precisa especificar, como no caso do acesso seqüencial, se a comunicação está liberada para a entrada ou para a saída. O "D" serve para os dois casos.

A linha 20 grava (WRITE) "P. SANTOS" no buffer #1. Como este programa está aberto para acesso direto, "P. SANTOS" permanecerá no buffer #1 até que o programa o envie para outro lugar.

A linha 30 faz justamente isto. Coloca (PUT) o conteúdo do buffer #1 registro 2 do arquivo de disco:



Neste ponto, "P. SANTOS" não está mais no buffer #1. Ele está no registro 2 do arquivo de disco.

A linha 34 pega (GET) o registro 2 e o grava no buffer #1. Agora, "P. SANTOS" está no arquivo de disco e no buffer #1.
A linha 36 introduz (INPUT) o conteúdo do buffer #1 na memória principal e o chama de A$. Agora, "P. SANTOS" está no arquivo de disco e na memória principal. Porém, não está mais no buffer #1.
Com "P. SANTOS" na memória principal, a linha 38 pode imprimi-lo.
Note que nosso desenho mostra apenas dois registros no arquivo.
Pegue (GET) o registro 4. Digite:

```
34 GET #1, 4
```

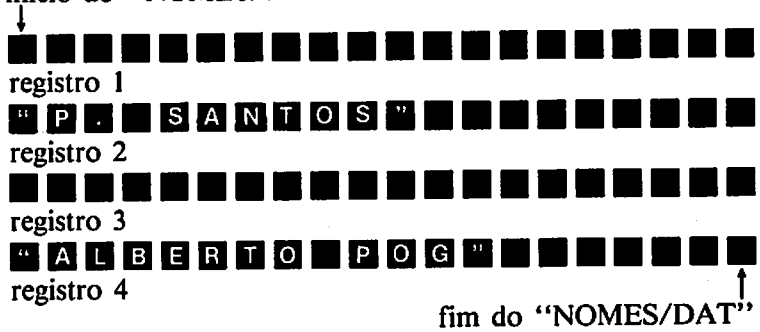
e rode o programa. O computador dá a você um erro IE (introdução de dados depois do fim do arquivo). O último registro que o programa coloca (PUT) no arquivo é o registro número 2. Ele se torna, portanto, o fim do arquivo.

Nota: Você não obteve este erro? Então você já deve ter um arquivo "NOMES/DAT" no seu disco com três ou mais registros.

Para colocar (PUT) mais registros no arquivo, acrescente estas linhas abaixo. Depois, rode o programa:

```
31 WRITE #1, "ALBERTO POG"
32 PUT #1,4
```

Agora, seu arquivo "NOMES/DAT" terá estes quatro registros:



TRABALHANDO COM INFORMAÇÕES INÚTEIS

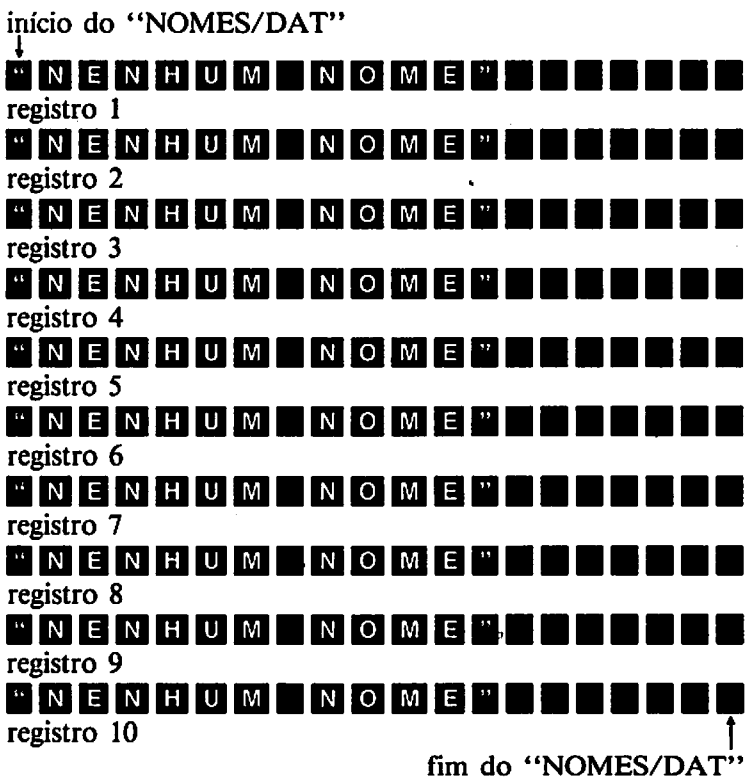
Você ainda não colocou (PUT) nada no registro 1. Peça ao computador para retirar (GET) o conteúdo do registro 1 e ver o que acontece. Digite isto e rode o programa:

```
34 GET #1, 1
```

Desde que o computador não tenha colocado (PUT) nada no registro 1, este contém apenas informações inúteis.
Quando você pedir ao computador para retirar (GET) e introduzir (INPUT) o conteúdo do registro 1, ele fornecerá aquelas "informações" ou apresentará a você um erro OS (insuficiência de espaço de string). O erro OS significa que as "informações inúteis" consomem mais de 200 bytes (caracteres).
Como os seus registros vazios contêm informações não desejadas, que tal colocarmos alguma informação mais útil neles? Limpe a memória e digite este programa:

```
10 OPEN "D", #1, "NOMES/DAT"
20 FOR X = 1 TO 10
30 WRITE #1, "NENHUM NOME"
40 PUT #1, X
50 NEXT X
60 CLOSE #1
```

Rode-o. Este programa monta um arquivo de disco chamado "NOMES/DAT" que tem 10 registros. Cada registro contém "NENHUM NOME":



Agora, limpe a memória e digite isto:

```
5 CLS
10 OPEN "D", #1,"NOMES/DAT"
15 PRINT
20 PRINT:INPUT "REGISTRO NO.(1-10)";R
30 IF R > 10 THEN 20
40 IF R < 1 THEN 130
```

```
50 GET #1,R
60 INPUT #1, A$
70 PRINT A$"— E' O NOME NO REGISTRO"R
80 INPUT"DIGITE O NOVO NOME OU PRESSIONE <ENTER>";A$
90 IF A$ = "" THEN 20
100 WRITE #1, A$
110 PUT #1,R
120 GOTO 20
130 CLOSE #1
```

Rode-o. Veja como todos os seus registros têm inicialmente "NENHUM NOME". Depois, você pode mudar os dados nos registros tantas vezes quanto quiser. (Para terminar o programa, digite um registro de nº 0.)

LENDO TODOS OS REGISTROS

Neste ponto, você gostaria que o computador imprimisse o conteúdo dos registros do seu arquivo "NOMES/DAT" juntamente com seu número de registro apropriado. Preserve o seu programa. Se você quiser, limpe a memória, digite-o e rode-o:

```
10 OPEN "D", #1, "NOMES/DAT"
20 R = 1
30 GET #1, A$
40 INPUT #1, A$
50 PRINT A$ "— ESTA' NO REGISTRO" R
60 IF R = 10 THEN 90
70 R = R + 1
80 GOTO 30
90 CLOSE #1
```

A linha 20 faz R igual a 1. Nas próximas linhas, o computador retira (GET), introduz (INPUT) e imprime (PRINT) o conteúdo do registro 1.
A linha 70 faz R igual a 2, e o processo todo se repete com o registro 2. Quando R é igual a 10 — o último registro no arquivo —, o programa termina.
Às vezes, não sabemos o número do último registro. Assim, altere a linha 60 e rode o programa:

```
60 IF R = LOF(1) THEN 90
```

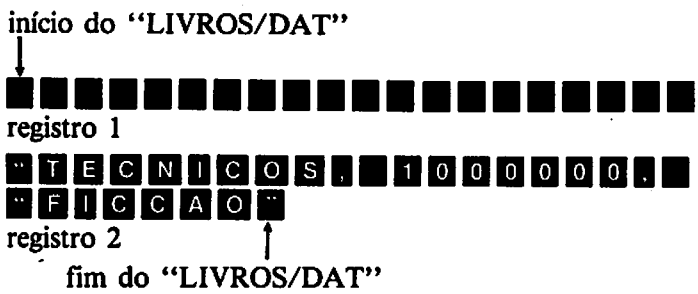
LOF observa o arquivo que está em comunicação com o buffer #1 (o número que está entre parênteses), e diz ao computador qual o número do último registro que está no arquivo.

UM REGISTRO MAIS PODEROSO

Até agora, nós estivemos colocando apenas um "campo" de dados em cada registro. Podemos organizar mais o arquivo, dividindo cada registro em vários campos. Limpe a memória, digite e rode este programa:

```
10 OPEN "D", #1, "LIVROS/DAT"
20 WRITE #1, "TECNICOS", 1000000, "FICCAO"
30 PUT #1, 2
34 GET #1, 2
36 INPUT #1, T$, N, F$
38 PRINT T$, N, F$
40 CLOSE #1
```

A linha 20 grava (WRITE) três campos de dados no buffer #1. Depois, a linha 30 coloca (PUT) o conteúdo inteiro do buffer #1 (todos os três campos) no registro 2 do arquivo:



A linha 34 retira (GET) todo o conteúdo do registro 2 e o grava no buffer #1. Depois a linha 36 introduz (INPUT) na memória todos os três campos de dados do buffer #1 e os rotula como T$, N e F$.
Substitua a linha 36 por:

```
36 INPUT #1, T$
```

e rode o programa.

Como esta linha pede ao computador para introduzir (INPUT) na memória o primeiro campo de dados do buffer #1, ele introduz (INPUT) apenas "TECNICOS".

FAÇA VOCÊ MESMO #6.1

O que você acha que o computador imprimiria se você rodasse o programa, substituindo a linha 36 pela de baixo? Por quê?

```
36 INPUT #1, N
```

FAÇA VOCÊ MESMO #6.2

Altere o programa que armazena o arquivo "NOMES/DAT" de modo que cada registro contenha cinco campos de dados:

1. *nome*
2. *endereço*
3. *cidade*
4. *Estado*
5. *CEP*

EXERCÍCIOS DO CAPÍTULO

1. *O que são registros? Por que você deve usá-los para acessar diretamente os dados?*
2. *O que são campos?*
3. *Qual é a diferença entre um arquivo de acesso seqüencial e um de acesso direto?*
4. *Por que a atualização de um arquivo de acesso direto é mais rápida?*

CAPÍTULO 7

a capacidade de um disco

Seu disco está dividido em milhares de unidades do mesmo tamanho. Cada unidade é chamada de "byte". Um byte pode representar um caractere. Assim, a palavra QUEIJO ocupa 6 bytes do disco.
Um disco vazio contém 161 280 bytes; 4 608 para o diretório e 156 672 para os arquivos de disco.

Nota: Um disco contém 35 trilhas. Cada trilha contém 18 setores de 256 bytes, ou 18 × 256 = 4 608 bytes. Uma das trilhas é reservada para o diretório. Sobram ainda 156 672 bytes (4 608 bytes por trilha × 34 trilhas).

Isto significa que você pode usar todos os 156 672 bytes para os dados? Possivelmente. Há dois fatores que determinam isto. O primeiro depende do modo como o computador reserva espaço para o arquivo de disco. Ele armazena o arquivo em grupos. (Nós os chamamos blocos.) Cada bloco contém 2 304 bytes.
Por causa disto, todos os seus arquivos de disco contêm um múltiplo de 2 304 bytes. Se seu arquivo contém 2 305 bytes de dados, por exemplo, o computador reservará dois blocos para ele, ou 4 608 bytes (2 304 × 2). É desta maneira que o computador reserva espaço de arquivo, pois é o modo mais eficiente de se criar um arquivo. Para alterar isso, é necessário conhecimento técnico. (Veja o Apêndice F, "Informações Técnicas".)
O segundo fator que influi na quantidade de dados que você pode colocar em um disco é o seu programa. Alguns programas são muito eficientes. Outros possuem uma grande quantidade de cabeçalho e espaço vazio no arquivo.
Nos próximos dois capítulos, iremos comparar oito diferentes tipos de programas. Cada um armazenará o mesmo dado — 8, "LARANJA", −79, e "BANANA" — em um arquivo de disco chamado "FEIRA/DAT". A quantidade de cabeçalho e espaços vazios que cada programa introduz em "FEIRA/DAT" variará enormemente.

GRAVANDO EM UM DISCO

O Programa 1 usa WRITE para colocar os dados no disco. Digite e rode:

PROGRAMA 1

26 bytes

```
10 OPEN "O", #1, "FEIRA/DAT"
20 WRITE #1, 8, "LARANJA"
30 WRITE #1, -79, "BANANA"
40 CLOSE #1
```

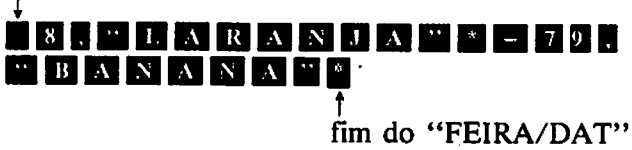
Há um modo fácil de ver o que as linhas 20 e 30 gravam no seu disco. Digite essas duas linhas, porém omita o #1 de cada uma. Isto impede o computador de gravar no disco (pelo buffer #1). Ele exibe os dados na tela. Digite:

WRITE 8, "LARANJA" ENTER
WRITE −79, "BANANA" ENTER

Veja com cuidado o que o computador gravou. Todos os espaços em branco e todas as pontuações.
Note o modo que o computador grava as duas strings (LARANJA e BANANA). Ele as exibe entre aspas e grava os números (8 e −79) de maneira diferente. Se o número é negativo, ele é precedido pelo sinal de menos. Se o número é positivo, um espaço em branco o precede.
Há dois caracteres que você digitou e o computador não exibiu na tela. São os dois caracteres ENTER que você digitou nas linhas WRITE. No lugar deles, o computador saltou uma linha para baixo:

```
8, "LARANJA"
OK
−79, "BANANA"
OK
```

Em um disco, o computador grava cada caractere ENTER exatamente como você os digitou. A ilustração a seguir mostra o que o Programa 1 grava (WRITE) no disco. (Nós usamos o asterisco para representar o caractere ENTER.)



Nota: Vamos ser mais precisos? O que o computador grava (WRITE) no disco são códigos binários. Cada caractere tem um código ASCII (veja Apêndice D) que o computador converte em um número binário.

Conte os caracteres. Não se esqueça do ENTER (representado por um asterisco), da vírgula, das aspas e do espaço em branco antes do número 8. O Programa 1 faz o "FEIRA/DAT" consumir 26 bytes.

O PONTO DE VISTA DO DISCO

Para ler "FEIRA/DAT", digite e rode este programa de entrada (limpe a memória primeiro):

PROGRAMA DE ENTRADA

```
10 CLS
20 OPEN "I", #1, "FEIRA/DAT"
30 IF EOF(1) = −1 THEN 80
40 INPUT #1, A, B$
50 PRINT:PRINT"ITEM:"A
```

```
60 PRINT"ITEM:"B$
70 GOTO 30
80 CLOSE #1
```

Seus itens de dados foram introduzidos. Mas, não as aspas, vírgulas e os espaços em branco intercalados entre seus dados.

COMO O ARQUIVO ESTÁ GRAVADO

Para ver o que o Programa 1 gravou no seu disco, você pode usar um programa com LINE INPUT. Primeiro, armazene (SAVE) o programa de entrada que temos na memória. (Nós o usaremos mais tarde.) Agora, troque-o por um programa LINE INPUT. Anule a linha 50 e altere as linhas 40 e 60. Digite:

```
40 LINE INPUT #1, L$
50
60 PRINT "LINHA DE DADOS :" L$
```

Rode-o... A linha 40 coloca (INPUT) na memória uma linha inteira, em vez de um único item de dado do arquivo de disco. Esta linha inclui tudo até o caractere ENTER — pontuação, espaço etc. No arquivo "FEIRA/DAT", a primeira linha contém 8,"LARANJA". A linha 40 chama esta linha de L$ e a linha 60 a exibe na tela.
O programa coloca (INPUT) na memória e imprime (PRINT) −79,"BANANA", a segunda e última linha do arquivo.

CONTAGEM DE BYTES DO ARQUIVO

Nós podemos alterar facilmente este programa de modo que ele conte quantos bytes estão no arquivo. Acrescente estas linhas e rode o programa:

```
25 PRINT "ESTE ARQUIVO CONTEM :"
27 PRINT: PRINT: PRINT: PRINT
57 M$ = L$ + "*"
60 PRINT M$;
65 L = LEN(M$) + L
90 PRINT@394, L "BYTES"
```

A linha 57 coloca um asterisco em cada linha (LINE). Este asterisco representa o caractere ENTER. A linha 65 conta o número total de caracteres (bytes) em cada linha.
Aqui estão os dois programas juntos:

PROGRAMA LINE INPUT

```
10 CLS
20 OPEN "I", #1, "FEIRA/DAT"
25 PRINT "ESTE ARQUIVO CONTEM :"
27 PRINT: PRINT: PRINT: PRINT
30 IF EOF(1) = -1 THEN 80
40 LINE INPUT #1, L$
57 M$ = L$ + "*"
60 PRINT M$
65 L = LEN(M$) + L
70 GOTO 30
80 CLOSE #1
90 PRINT@394, L "BYTES"
```

Armazene-o. Ele será útil quando formos comparar todos os programas.

PRINT — PARA VARIAR

Até agora, nós apenas usamos WRITE para colocar dados em um arquivo de disco. Se você já programou em BASIC, deve estar acostumado a utilizar PRINT ao invés de WRITE.
O sistema de disco do CP 400 permite sua utilização. Entretanto, PRINT é muito mais complicado. Senão, vejamos. Elimine (KILL) seu velho arquivo "FEIRA/DAT", digitando:

KILL "FEIRA/DAT" ENTER

Agora, limpe a memória, digite e rode o Programa 2. Depois, rode o Programa INPUT ou INPUT LINE, aquele que você preferir.

PROGRAMA 2

47 Bytes

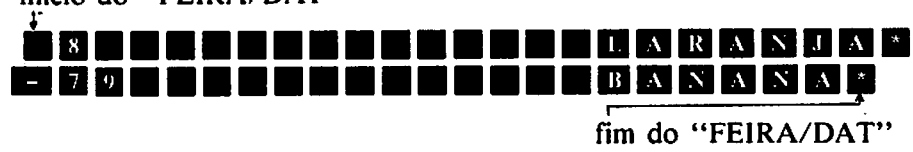
```
10 OPEN "O", #1, "FEIRA/DAT"
20 PRINT #1, 8, "LARANJA"
30 PRINT #1, -79, "BANANA"
40 CLOSE #1
```

As linhas 20 e 30 imprimem (PRINT) seus dados no buffer #1 que, como você já sabe, está em um dos 15 buffers que enviarão seus dados ao arquivo de disco. Para ver o que o Programa 2 imprime (PRINT), digite:

PRINT 8, "LARANJA" ENTER
PRINT -79, "BANANA" ENTER

Observe que o computador não coloca as strings (LARANJA e BANANA) entre aspas, como acontece com a WRITE. Esse é um dado muito importante.

Agora, olhe para os espaços em branco. O espaço antes do 8 tem o mesmo significado do espaço em branco que vimos na instrução WRITE, isto é, indica que ele é um número positivo.
Agora, os outros espaços em branco... Depois do número o computador imprime uma série de espaços em branco. Por quê?
Veja o que a vírgula faz numa linha PRINT. (Essa informação pode ser encontrada no manual do CP 400.) Ela faz com que o computador imprima seus dados em colunas, inserindo espaços entre elas.
O computador imprime cada um destes espaços em branco no seu arquivo de disco:



Conte todos os caracteres. O Programa 2 faz o "FEIRA/DAT" consumir 47 bytes.

ECONOMIZANDO ESPAÇO

Os espaços em branco que PRINT insere no seu arquivo de disco são um desperdício de área para armazenamento de informações. Um jeito de evitar esse desperdício consiste no uso do ponto e vírgula. Veja no manual como um ponto e vírgula comprime seus dados. Digite:

```
PRINT 8; "LARANJA" ENTER
PRINT -79; "BANANA" ENTER
```

Você pode comprimir seus dados no disco da mesma forma. Limpe a memória e elimine seu arquivo "FEIRA/DAT". Depois, digite e rode este programa:

PROGRAMA 3

22 bytes

```
10 OPEN "O", #1, "FEIRA/DAT"
20 PRINT #1, 8; "LARANJA"
30 PRINT #1, -79; "BANANA"
40 CLOSE #1
```

Isto é o que o Programa 3 imprime em seu disco. (Para testá-lo use o programa LINE INPUT.)

Apenas 22 bytes. Há somente três espaços em branco neste arquivo de disco. Há um espaço antes do 8 (para indicar que é positivo) e espaços depois do 8 e – 79 (para indicar que eles são números). Não há espaços em branco nos extremos das strings.

ALGUNS TRUQUES

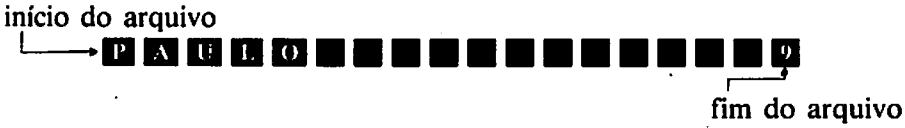
Há certos tipos de linhas PRINT que são cheias de truques. Digite:

```
PRINT #1,"LARANJA"; "BANANA" ENTER
PRINT #1, "VERDE, AMARELO" ENTER
PRINT #1, "PAULO", 9 ENTER
```

A linha PRINT #1, "LARANJA"; "BANANA" (no seu programa de disco) imprimiria no seu arquivo de disco:

O computador leria LARANJABANANA na memória como um único item. (Razão: não há um delimitador — uma vírgula, aspa ou espaço — para separar LARANJA de BANANA.)
A linha PRINT #1, "VERDE, AMARELO" imprimiria isto em seu arquivo de disco:

O computador leria VERDE, AMARELO como dois itens: VERDE e AMARELO. (Razão: o computador interpreta a vírgula como um delimitador.)
A linha PRINT #1, "PAULO", 9, imprimiria isto em seu arquivo de disco:



O computador leria PAULO 9 (com todos os espaços em branco) na memória como apenas um item. (Razão: apesar do computador normalmente interpretar espaços em branco como um delimitador, ele não os interpreta dessa forma quando seguem uma string ou antecedem um número.)

UM ARQUIVO DE DISCO ATRAENTE

PRINT USING é outra palavra que você pode substituir por WRITE. Nós discutimos PRINT USING no manual do CP 400.

```
PRINT USING "%    %$+ # #.# #"; "LARANJA", 8 ENTER
PRINT USING "%    %$+ # #.# #"; "BANANA", -79 ENTER
```

Você pode fazer com que o computador imprima estas mesmas imagens em seu disco com este programa. Elimine "FEIRA/DAT", limpe a memória, digite e rode o programa abaixo:

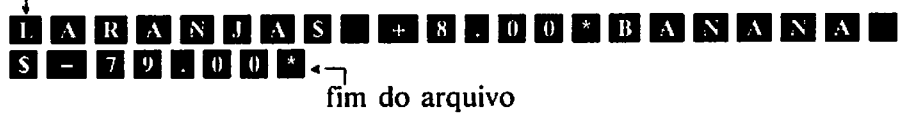
PROGRAMA 4

30 bytes

```
10 OPEN "O", #1, "FEIRA/DAT"
20 PRINT #1, USING"%    %$+ # #.# #"; "LARANJA", 8
30 PRINT #1, USING"%    %$+ # #.# #"; "BANANA", -79
40 CLOSE #1
```

Esse programa imprime isso em seu arquivo de disco:

início do arquivo



Nota: Há 5 espaços em branco entre os caracteres % nas linhas 20 e 30. Contando os dois caracteres %, este campo de string (para imprimir LARANJA e BANANA) contém sete bytes.

Agora, os dados já estão com um formato de impressão bem melhor. Você pode introduzi-los e imprimi-los usando o programa LINE INPUT. Todos os arquivos que você criou neste capítulo são de acesso seqüencial. O próximo capítulo compara mais 4 programas que colocam os mesmos dados em arquivos de acesso direto.

EXERCÍCIOS DO CAPÍTULO

1. *Qual é o menor tamanho de um arquivo em disco? Por que ele não pode ser menor?*
2. *Como o computador grava (WRITE) números em um arquivo de disco?*
3. *Como ele grava (WRITE) strings?*
4. *Qual é a diferença entre INPUT e LINE INPUT?*
5. *O que o computador faz com uma vírgula em um PRINT?*
6. *O que o computador faz com um ponto e vírgula em uma linha PRINT?*
7. *Como o computador imprime (PRINT) strings?*

CAPÍTULO 8

aparando as arestas

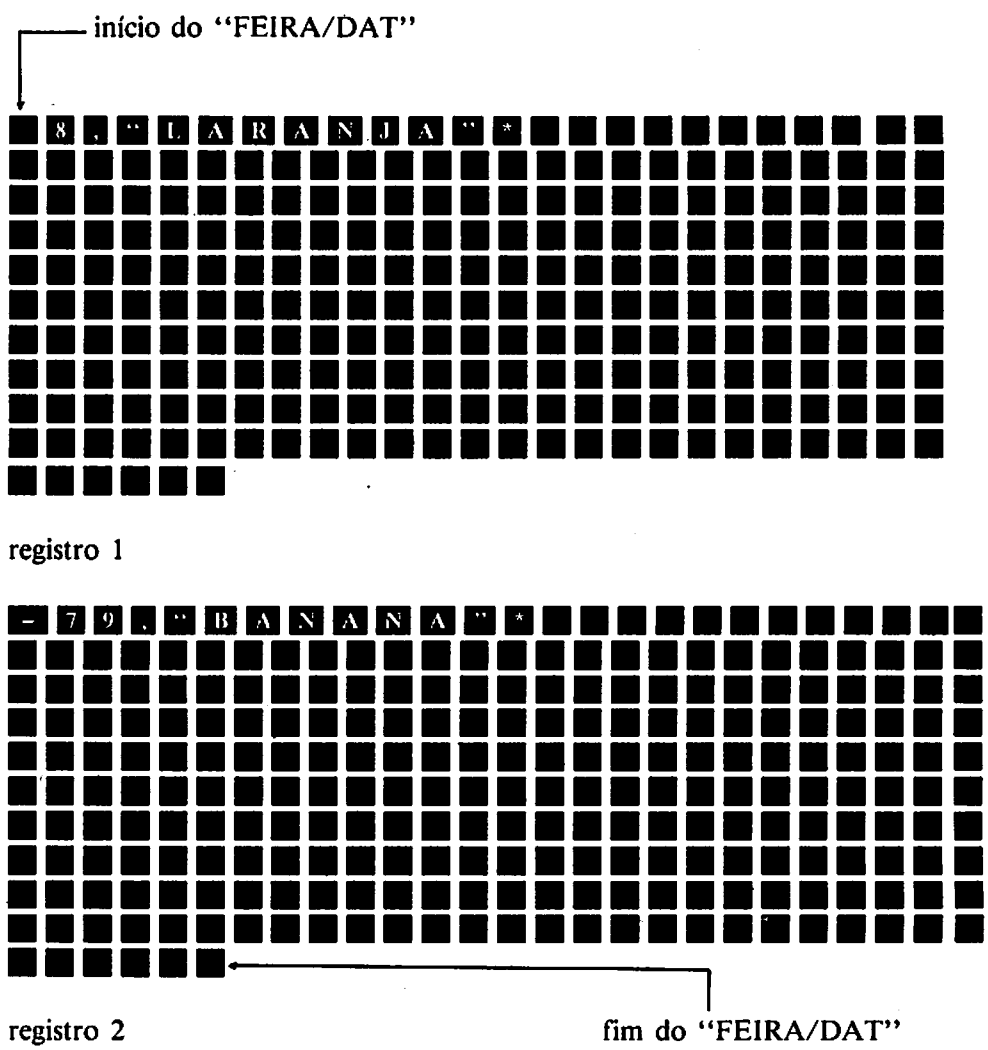
Geralmente os arquivos de acesso direto contêm uma grande quantidade de espaços vazios. Por exemplo, nosso primeiro programa é muito parecido com o Programa 1 do último capítulo. As linhas WRITE são idênticas. Mas, por ser de acesso direto, faz com que o "FEIRA/DAT" consuma 512 bytes:

PROGRAMA 5

512 bytes

```
10 OPEN "D", #1, "FEIRA/DAT"
20 WRITE #1, 8, "LARANJA"
30 PUT #1, 1
40 WRITE #1, -79, "BANANA"
50 PUT #1, 2
60 CLOSE #1
```

Um programa de acesso direto coloca dados dentro dos registros. Cada registro contém 256 bytes. O Programa 5 coloca dois registros no arquivo "FEIRA/DAT". Assim, ele consome 2 × 256, ou 512 bytes:



É um grande desperdício de espaço. Note que o que o computador grava em cada registro:

8,"LARANJA"*
-79,"BANANA"*

é o mesmo que o Programa 1 gravou. Conte os bytes. Treze bytes no primeiro registro e treze no segundo. Essa informação nos será útil quando estruturarmos nosso próximo programa.

Nota: Nós poderíamos ter usado PRINT ou PRINT USING ao invés de WRITE. O computador teria então imprimido seus dados dentro de cada registro usando o formato PRINT ou o PRINT USING.

SOFISTICANDO

O Programa 6 é idêntico ao 5, porém possui um número 13 no fim da linha 10. Isto diz ao computador para formar registros de 13 bytes de comprimento:

PROGRAMA 6

26 bytes

```
10 OPEN"D", #1,"FEIRA/DAT",13
20 WRITE #1,8,"LARANJA"
30 PUT #1,1
40 WRITE #1,-79,"BANANA"
50 PUT #1,2
60 CLOSE #1
```

que realmente reduz este arquivo:

início do "FEIRA/DAT"

8 , " L A R A N J A " *

registro 1

- 7 9 , " B A N A N A " * ← fim do "FEIRA/DAT".

registro 2

Em um arquivo de acesso direto, todos os registros devem ter o mesmo comprimento. (Consulte o Capítulo 6 para obter informações mais detalhadas.) Se você não especifica o comprimento, o computador entende que ele é de 256 bytes.
Neste programa, fizemos os registros com o tamanho do maior. (Por coincidência, no nosso programa os dois têm 13 bytes.) Digite e rode o Pro-

grama 6, se quiser. (Certifique-se de que a memória esteja limpa e elimine o seu arquivo anterior "FEIRA/DAT".) Depois de rodar o Programa 6, você poderá usá-lo para introduzir o arquivo:

PROGRAMA DE ENTRADA DIRETA

```
10 OPEN"D", #1,"FEIRA/DAT",13
20 R=R+1
30 GET #1,R
40 INPUT #1,A,B$
50 PRINT"REGISTRO"R":"A;B$
60 IF LOF(1)<>R THEN 20
70 CLOSE #1
```

Notas: 1 — Você não pode usar o programa LINE INPUT para determinar quantos bytes este arquivo consome. LINE INPUT não introduz os espaços que seguem o caractere ENTER em um registro.
2 — LOF — número do último registro gravado em arquivo direto.

MAIS EFICIÊNCIA

Um programa pode ser muito mais eficiente. Nosso próximo programa de acesso direto consome apenas 20 bytes. Limpe a memória, elimine o arquivo "FEIRA/DAT" anterior, digite e rode o Programa 7.

PROGRAMA 7

20 bytes

```
10 OPEN "D", #1,"FEIRA/DAT",10
20 FIELD #1,3 AS A$, 7 AS B$
30 LSET A$ = "8"
40 LSET B$ = "LARANJA"
50 PUT #1, 1
60 LSET A$ = "-79"
70 LSET B$ = "BANANA"
80 PUT #1, 2
90 CLOSE #1
```

A linha 20 faz com que o computador divida cada registro em dois campos. O primeiro campo é A$; e o segundo, B$. Estes dois campos terão o mesmo tamanho em cada registro. A$ sempre terá 3 bytes e B$, sempre 7. Agora que já combinamos isso, podemos colocar dados em cada campo. A linha 30 coloca (SET) o caractere 8 à esquerda (LEFT) de A$. Como o caractere 8 consome apenas 1 byte e há 3 bytes no campo de A$, ficarão dois espaços vazios depois do 8. Note que temos de converter o número 8 em uma string, colocando-o entre aspas. Você não pode justificar um número à esquerda. Você deve convertê-lo em uma string. A linha 40 justifica a palavra LARANJA à esquerda no campo B$.

A linha 50 coloca tudo isto no registro 1. Depois, o mesmo processo se repete para o registro 2.
Rode este programa para armazenar seu novo arquivo "FEIRA/DAT" no disco. Usando FIELD e LSET, seu programa funcionará do mesmo modo que o programa de acesso direto. A diferença é que FIELD e LSET colocam em cada registro somente o essencial.

início do "FEIRA/DAT"

registro 1

registro 2

Armazene este programa no disco pois você vai usá-lo mais tarde.

PROGRAMA DE ENTRADA COM CAMPO

```
10 OPEN "D", #1, "FEIRA/DAT", 10
20 FIELD #1, 3 AS A$, 7 AS B$
30 R = R + 1
40 GET #1, R
50 PRINT "REGISTRO" R ":" A$; B$
60 IF LOF(1) <> R THEN 30
70 CLOSE #1
```

Vamos ver agora o "Programa de Entrada com Campo". Note que usamos uma linha com FIELD. Rode o programa sem a linha 20 e veja o que acontece...
Sem a linha de FIELD, o computador não sabe onde estão os dois campos. Sempre que você introduzir registros com campo, use uma linha de FIELD em seu programa de entrada.
Você pode adivinhar o que o computador faria se você tentasse justificar uma string longa, tal como "123456789", à esquerda em um dos campos? Carregue o Programa 7, troque a linha 30 e rode o programa. (Primeiro, preserve (SAVE) o "Programa de Entrada com Campo". Não se esqueça da linha 20.)
30 LSET A$ = "123456789"
Agora, carregue e rode o "Programa de Entrada com Campo".
A$ tem apenas 3 bytes. Assim, o computador justifica à esquerda apenas os primeiros 3 bytes de "123456789" e elimina os caracteres restantes:

início do "FEIRA/DAT"

registro 1

registro 2
Posteriormente, voltaremos a esse assunto...

FAÇA VOCÊ MESMO #8.1

Escreva um programa de acesso direto para colocar uma relação de nomes e endereços em um arquivo de disco. Faça um registro de 57 bytes com estes seis campos:

1. último nome — 15 bytes
2. primeiro nome — 10 bytes
3. endereço — 15 bytes
4. cidade — 10 bytes
5. Estado — 2 bytes
6. CEP — 5 bytes

FAÇA VOCÊ MESMO #8.2

Escreva um programa para ler o arquivo que você criou no "Faça você mesmo" #8.1.

UM NÚMERO É UM NÚMERO

Vamos supor que você coloque vários números em seu arquivo de disco. Cada número deve ter um comprimento diferente:

-89.345683 68 385678

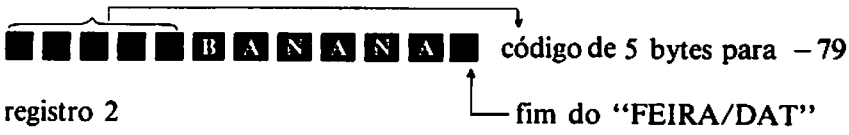
É muito importante que o computador não elimine nenhum dígito. Isto pode modificar o valor do número.
A palavra MKN$ resolverá este problema:

PROGRAMA 8

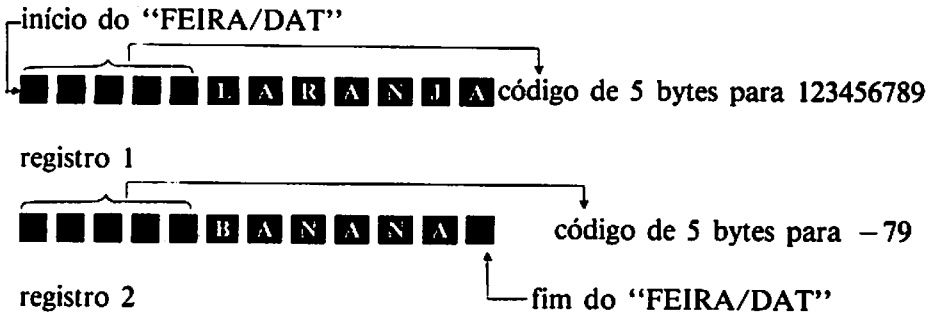
24 bytes

```
10 OPEN "D", #1, "FEIRA/DAT", 12
20 FIELD #1, 5 AS A$, 7 AS B$
30 LSET A$ = MKN$(8)
40 LSET B$ = "LARANJA"
50 PUT #1, 1
60 LSET A$ = MKN$(-79)
70 LSET B$ = "BANANA"
80 PUT #1, 2
90 CLOSE #1
```

A diferença entre este programa e o Programa 7 está nas linhas 10, 20, 30 e 60. Isto é o que ele armazena em seu arquivo de disco.



MKN$ converte um número em uma string codificada. Indiferente ao comprimento do número, MKN$ sempre converterá o número em uma string de 5 bytes de comprimento.
Por exemplo, altere a linha 30 de modo que ela justifique à esquerda um número de mais de 5 dígitos. Limpe a memória, elimine "FEIRA/DAT", digite e rode o programa. Isto é o que ele armazena em seu arquivo de disco:



Para ler este programa, você precisa decodificar a string. Carregue o "Programa de Entrada com Campo" e faça estas mudanças nele:

```
10 OPEN "D", #1, "FEIRA/DAT", 12
20 FIELD #1, 5 AS A$, 7 AS B$
50 PRINT "REGISTRO" R ":"; CVN(A$); B$
```

e rode-o... CVN (na linha 50) converte A$ no número que ele representa.

Nota: O computador vê somente os 9 primeiros dígitos de um número. Ele arredonda o resto.

FAÇA VOCÊ MESMO #8.3

Escreva um programa de acesso direto com campo que armazena a população de todos os Estados brasileiros. Faça cada registro conter 15 bytes, com estes dois campos:
1. Estado — 10 bytes
2. população — 5 bytes

FAÇA VOCÊ MESMO #8.4

Escreva um programa que introduza o arquivo que você criou no "Faça você mesmo" #8.3.

EXERCÍCIOS DO CAPÍTULO

1. *Se você não especificar o comprimento de um registro, quantos bytes o registro conterá?*
2. *Por que você deve incluir uma linha FIELD quando você justifica seu dado à esquerda (LSET)?*
3. *Em quantos bytes MKN$ converterá um número?*

CAPÍTULO 9

associando os arquivos de disco

Armazenar e recuperar arquivos em disco é tão fácil que você vai querer usá-los tantas vezes quanto puder. Neste capítulo, discutiremos algumas de suas aplicações.

FUSÃO DE ARQUIVOS DE PROGRAMAS

Com este método, você pode construir um programa a partir de "módulos" de programa preservados (SAVE) no disco. Você pode então combinar ou fundir (MERGE) qualquer destes arquivos de programa com um programa que você tenha na memória.
Digite e armazene estes dois programas abaixo:

```
10 REM CONVERSAO DA IDADE EM MESES
20 N = N * 12
30 A$ = STR$(N) + "MESES"
SAVE "MESES/IDADE", A ENTER
```

```
10 REM CONVERSAO DA IDADE EM SEMANAS
20 N = N * 52
30 A$ = STR$(N) + "SEMANAS"
SAVE "SEMANAS/IDADE", A ENTER
```

Certifique-se de digitar o A ao preservar (SAVE) estes programas. Nós explicaremos as razões deste cuidado mais tarde. Limpe a memória e digite as linhas:

```
5 INPUT "DIGITE SUA IDADE"; N
40 PRINT "VOCE VIVEU APROXIMADAMENTE" A$
```

e combine-as com um dos programas armazenados no disco, digitando:

```
MERGE "MESES/IDADE" ENTER
```

Liste o programa... O computador funde "MESES/IDADE" com o programa que ele tem na memória. Note que os números das linhas continuam os mesmos. O programa na memória fica assim:

```
5 INPUT "DIGITE SUA IDADE"; N
10 REM CONVERSAO DA IDADE EM MESES
20 N = N * 12
30 A$ = STR$(N) + "MESES"
40 PRINT "VOCE VIVEU APROXIMADAMENTE" A$
```

Agora digite:
MERGE "SEMANAS/IDADE" ENTER para fundir este programa com o programa "SEMANAS/IDADE". Depois, liste o programa inteiro. Observe que as linhas 10, 20 e 30 do programa que você tinha na memó-

ria foram substituídas pelas linhas 10, 20 e 30 do programa "SEMANAS/IDADE". Os números das linhas dizem ao computador como concatenar os dois programas. Se os dois números de linha coincidem (por exemplo, duas linhas 10), a linha do arquivo de disco prevalece.

Agora, um pouco de conhecimento técnico (para aqueles que se interessam). O computador normalmente grava o seu arquivo de dados no disco com o código ASCII de cada caractere. Por exemplo, ele grava a palavra OU com dois códigos — código ASCII de "O" (79) e o código ASCII de "U" (85). (Os códigos ASCII estão listados no Apêndice D.)

Entretanto, ao preservar um programa em disco, o computador gravará as palavras do BASIC de modo diferente, comprimindo-as num código "binário" de um byte.

Você não pode fundir um programa que contenha estes códigos binários. Foi por isso que você digitou a letra A ao armazenar os dois programas acima. O A diz ao computador para gravar os códigos ASCII de todas as palavras do BASIC em vez do código binário.

O diretório lhe diz se os dados dos seus arquivos estão em código ASCII ou binário. A letra A na quarta coluna indica que o programa está todo em código ASCII, enquanto a B, que algumas palavras estão em código binário.

Nota: Tente digitar MERGE "MESES/IDADE", R ENTER. O R diz ao computador para rodar o programa depois da fusão.

USANDO MAIS ESPAÇOS DE BUFFER

Quando você inicializa seu sistema de disco, ele separa duas áreas de buffer na memória para comunicação de disco. Você pode usar uma ou ambas as áreas para leitura ou gravação de dados de um arquivo em disco. Até agora só utilizamos os buffers #1 e #2. Mas, como dissemos anteriormente, você pode usar até 15 áreas de buffer de disco.

Para utilizar mais do que dois buffers, você deve, em primeiro lugar, reservar espaço na memória para eles. Para fazer isso, use a palavra FILES. Por exemplo, FILES 3 reserva 3 buffers.

O uso destes buffers simplificará muito seus programas. Por exemplo, vamos supor que você seja dono de uma escola de computação e deseja organizá-la. O primeiro passo consiste em introduzir todos os alunos em um arquivo chamado "COMPUTADOR/ESC". Limpe a memória, digite e rode o programa abaixo:

```
10 OPEN "O", #1, "COMPUTADOR/ESC"
20 FOR X=1 TO 6
30 READ A$
40 PRINT #1, A$
50 NEXT X
```

```
60 CLOSE #1
70 DATA JOAO, MARIA, JOSE
80 DATA PAULO, ROSA, RITA
```

Agora, você pode gravar este programa que distribui os alunos em classes de linguagem BASIC ou Assembly. Limpe a memória e digite este programa:

PROGRAMA DE ORGANIZAÇÃO DE CLASSES

```
10 FILES 3
20 OPEN "O", #1, "BASIC/CLS"
30 OPEN "O", #2, "ASSEMBLY/CLS"
40 OPEN "I", #3, "COMPUTADOR/ESC"
50 IF EOF(3) = -1 THEN 120
60 INPUT #3, ALN$
70 PRINT: PRINT ALN$
80 INPUT "LINGUAGEM (1) BASIC OU (2) ASSEMBLY"; R
90 IF R>2 THEN 80
100 WRITE #R, ALN$
110 GOTO 50
120 CLOSE #1
130 CLOSE #2
140 CLOSE #3
```

Rode-o. Depois de distribuir todos os alunos em classes, você pode imprimir uma lista de chamada com este programa. Limpe a memória, digite e rode o programa abaixo:

PROGRAMA DE LISTA DE CHAMADA

```
10 CLS
20 PRINT "BASIC/CLS" : PRINT
30 OPEN "I", #1, "BASIC/CLS"
40 IF EOF(1) = -1 THEN 80
50 INPUT #1, A$
60 PRINT A$
70 GOTO 40
80 CLOSE #1
```

Nota: Substitua "ASSEMBLY/CLS" por "BASIC/CLS" nas linhas 20 e 30 para imprimir a lista de chamada da classe de linguagem Assembly.

O "Programa de Organização de Classes" tem três buffers abertos ao mesmo tempo. Por isso, é possível se comunicar com três arquivos de disco ao mesmo tempo.
A linha 10 reserva memória para estes três buffers. As linhas de 20 a 40 abrem (OPEN) os três buffers. Assim, a linha 60 introduz (INPUT) um aluno do arquivo "COMPUTADOR/ESC" no buffer #3.

A linha 100 grava (WRITE) o nome do aluno no buffer #1 ("BASIC/CLS") ou no buffer #2 ("ASSEMBLY/CLS").
Após a introdução de todos os alunos no buffer #3 ("COMPUTADOR/ESC"), a linha 50 desvia o programa para as linhas 120-140, que fecham (CLOSE) os três buffers.

BUFFER LOTADO

Há mais uma coisa a respeito de FILES. Limpe a memória, digite e rode:

```
10 CLEAR 400
20 FILES 1, 400
30 A$ = "NORMALMENTE VOCE NAO PODERA COLOCAR TODO ESSE TEXTO EM UM ARQUIVO DE DISCO AO MESMO TEMPO."
40 B$ = "SEM USAR FILES, VOCE APENAS TERA UM TOTAL DE 256 BYTES DE ESPACO DE BUFFER."
50 C$ = "NESTE PROGRAMA, RESERVAMOS 400 BYTES DE ESPACO DE BUFFER."
60 D$ = "DESSA FORMA, VOCE PODE ENVIAR TODO ESSE TEXTO AO BUFFER, AO MESMO TEMPO."
70 E$ = "ELE O ENVIARA PARA O ARQUIVO DE DISCO IMEDIATAMENTE."
80 OPEN "O", #1, "PALAVRA/DAT"
90 WRITE #1, A$, B$, C$, D$, E$
100 CLOSE #1
200 OPEN "I", #1 "PALAVRA/DAT"
210 INPUT #1, A$, B$, C$, D$, E$
220 CLS
230 PRINT A$; B$; C$; D$; E$
240 CLOSE #1
```

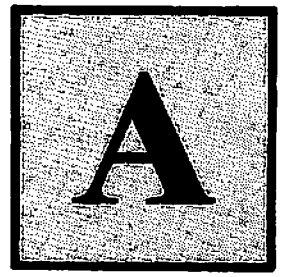
Nota: Você pode fazer o buffer do tamanho que desejar.

EXERCÍCIOS DO CAPÍTULO

1. *Como você armazena (SAVE) um programa que será concatenado (MERGE) com outro?*
2. *O que acontecerá quando os dois programas que você está combinando tiverem duas linhas com o mesmo número?*
3. *Quantos buffers o computador reserva quando ele é acionado?*
4. *Qual o tamanho do buffer reservado?*
5. *O que significa FILES 3,3000?*

APÊNDICES

RESPOSTAS DOS EXERCÍCIOS DE PROGRAMAÇÃO



FAÇA VOCÊ MESMO # 5.1

```
5 CLS
10 PRINT@0*32+7,"*** AMIGOS***"
20 PRINT@4*32+2,"VOCE JA CRIOU SEU
ARQUIVO"
30 PRINT@5*32+7,"'AMIGOS/DAT'
<S/N>"
40 INPUT A$
50 IF A$="S" THEN 500
55 CLS
60 PRINT@2*32+7,"CRIAR ARQUIVO"
80 OPEN"O", #1, "AMIGOS/DAT"
90 GOSUB 1000
100 IF N$="" THEN CLOSE#1:GOTO500
110 WRITE#1,N$,E$,T$
120 GOTO 90
500 CLS
510 PRINT@1*32+6,"ESCOLHA SUA OPCAO"
520 PRINT@3*32+8,"<1> LER"
530 PRINT@4*32+8,"<2> ALTERAR"
540 PRINT@5*32+8,"<3> ACRESCENTAR"
550 PRINT@6'32+8,"<4> ELIMINAR"
555 PRINT@7*32+8,"<5> FIM"
560 INPUT N
570 ON N GOTO 2000, 3000, 4000, 5000, 6000
1000 CLS: PRINT"PRESSIONE<ENTER>
QUANDO TERMINAR"
1010 INPUT"NOME: ";N$
1015 IF N$="" THEN 1040
1020 INPUT"ENDERECO: ";E$
1030 INPUT"TELEFONE: ";T$
1040 RETURN
2000 CLS
2010 OPEN "I",#1, "AMIGOS/DAT"
2020 IF EOF(1)=-1 THEN CLOSE#1:GOTO
500
2030 INPUT#1, N$, E$, T$
2040 PRINT "NOME: ";N$
2050 PRINT "ENDERECO: "; E$
2060 PRINT "TELEFONE: "; T$
2070 PRINT
2075 FORI=1TO1500:NEXT
2080 GOTO 2020
3000 CLS
3010 OPEN"I",#1,"AMIGOS/DAT"
3020 OPEN"O",#2,"ALTER/DAT"
3030 IF EOF(1)=-1 THEN 3800
3040 INPUT#1, N$, E$, T$
3050 PRINT:PRINT"NOME: ";N$
3060 PRINT"ENDERECO: ";E$
3070 PRINT"TELEFONE: ";T$
3080 PRINT"VAI MUDAR ALGUM ITEM"
3090 INPUT"<S/N>";X$
3100 IF X$="N" THEN 3700
3110 PRINT:PRINT"QUAL:"
3120 PRINT"<1> ENDERECO"
3130 PRINT"<2> TELEFONE"
3140 INPUT N
3150 ON N GOTO 3550, 3570
3550 INPUT"INTRODUZA O NOVO
ENDERECO";E$
3560 GOTO 3700
3570 INPUT"INTRODUZA O NOVO
TELEFONE";T$
3580 GOTO 3700
3700 WRITE#2,N$,E$,T$
3710 GOTO3030
3800 CLOSE#1,#2
3810 KILL"AMIGOS/DAT"
3820 RENAME"ALTER/DAT" TO
"AMIGOS/DAT"
3830 GOTO500
4000 CLS
4010 OPEN"I",#1,"AMIGOS/DAT"
4020 OPEN"O",#2,"ACRES/DAT"
4030 IF EOF(1)-1 THEN 4100
4040 INPUT#1,N$,E$,T$
4045 WRITE#2,N$,E$,T$
4050 GOTO4030
4100 PRINT"PRESSIONE<ENTER>QUANDO
TERMINAR":PRINT
4110 INPUT"NOME: ";N$
4120 IF N$="" THEN 4170
4130 INPUT"ENDERECO: ";E$
4140 INPUT"TELEFONE: ";T$
4150 WRITE#2,N$,E$,T$
4160 PRINT:GOTO4110
4170 IF N$="" THEN CLOSE#1,#2
4180 KILL"AMIGOS/DAT"
```

```
4190 RENAME "ACRES/DAT" TO
"AMIGOS/DAT"
4200 GOTO500
5000 CLS
5010 OPEN"I",#1,"AMIGOS/DAT"
5020 OPEN"O",#2,"ANUL/DAT"
5030 IF EOF(1)=-1 THEN 5200
5040 INPUT#1,N$,E$,T$
5045 PRINT
5050 PRINT"NOME: ";N$
5060 PRINT"ENDERECO: ";E$
5070 PRINT"TELEFONE: ";T$
5080 INPUT"QUER ANULAR ESTE<S/N>";R$
5090 IF R$="S" THEN 5030
5100 WRITE#2,N$,E$,T$
5110 GOTO5030
5200 CLOSE#1,#2
5210 KILL"AMIGOS/DAT"
5220 RENAME "ANUL/DAT" TO
"AMIGOS/DAT"
5230 GOTO500
6000 END
```

FAÇA VOCÊ MESMO # 6.1

Isto produz um erro FD — Dados Defeituosos em Arquivo — na linha 36. O primeiro campo no registro 2 é "TECNICOS", uma string. A linha 36 o introduz em N, uma variável numérica.

FAÇA VOCÊ MESMO # 6.2

```
5 CLS
10 OPEN"D",#1,"NOMES/DAT"
20 PRINT"<1> PARA ARMAZENAR"
25 PRINT"<2> PARA LER OU ALTERAR"
27 INPUT N
28 ON N GOTO 30,70
30 FOR X=1 TO 10
40 PRINT:PRINT"REGISTRO"X
50 GOSUB 180
60 NEXT X
65 GOTO170
70 INPUT"QUAL REGISTRO(1 - 10)";X
80 IF X>10 THEN 170
90 IF X<1 THEN END
100 GET#1,X
110 INPUT#1,N$,E$,C$,S$,Z$
120 PRINT:PRINT"REGISTRO"X
130 PRINT N$,E$,C$,S$,Z$
140 INPUT"VOCE QUER MUDAR ISTO
<S/N>"; R$
150 IF R$="S" THEN GOSUB 180
160 GOTO 70
170 CLOSE#1:END
180 INPUT"NOME: ";N$
190 INPUT"ENDERECO:";E$
200 INPUT"CIDADE:";C$
210 INPUT"ESTADO:";S$
220 INPUT"CEP:";Z$
230 WRITE#1,N$,E$,C$,S$,Z$
240 PUT#1,X
250 RETURN
```

FAÇA VOCÊ MESMO # 8.1

```
10 OPEN"D",#1,"REL/DAT",57
20 FIELD #1,10 AS UN$,15 AS PN$, 15 AS EN$,
10 AS CID$, 2 AS EST$, 5 AS CEP$
30 R=R+1
40 CLS
50 INPUT"ULTIMO NOME";U$
60 INPUT"PRIMEIRO NOME";P$
70 INPUT"ENDERECO";E$
80 INPUT"CIDADE";C$
90 INPUT"ESTADO";S$
100 INPUT"CEP";Z$
110 LSET UN$=U$
120 LSET PN$=P$
130 LSET EN$=E$
140 LSET CID$=C$
150 LSET EST$=S$
160 LSET CEP$=Z$
170 PUT #1,R
180 PRINT
190 INPUT"MAIS DADOS(S/N)";MD$
200 IF MD$="S" THEN 30
210 CLOSE#1
```

FAÇA VOCÊ MESMO # 8.2

```
10 OPEN"D",#1,"REL/DAT",57
20 FIELD#1,10 AS UN$,15 AS PN$,15 AS EN$,
10 AS CID$,2 AS EST$,5 AS CEP$
30 R=R+1
40 CLS
50 GET#1,R
60 PRINTUN$","PN$
70 PRINTEN$
80 PRINTCID$","EST$
90 PRINTCEP$
100 PRINT
110 IF LOF(1)=R THEN 140
```

```
120 INPUT "PRESSIONE<ENTER>PARA O
PROXIMO NOME";E$
130 GOTO30
140 CLOSE#1
```

FAÇA VOCÊ MESMO # 8.3

```
10 OPEN "D",#1, "POP",15
20 FIELD#1,10 AS ESTADO$, 5 AS POP$
30 R = R + 1
40 CLS
50 INPUT "ESTADO";E$
60 INPUT "POPULACAO";P
70 LSET ESTADO$ = E$
80 LSET POP$ = MKN$(P)
85 PUT#1,R
90 PRINT
100 INPUT "MAIS DADOS(S/N)"; MD$
110 IF MD$ = "S" THEN 30
120 CLOSE#1
```

FAÇA VOCÊ MESMO # 8.4

```
10 OPEN "D", #1, "POP", 15
20 FIELD#1, 10 AS ESTADO$, 5 AS POP$
30 R = R + 1
40 GET#1, R
50 PRINT ESTADO$, CVN (POP$)
60 IF LOF(1)< >R THEN 30
70 CLOSE#1
```

RESPOSTAS DAS VERIFICAÇÕES DOS CAPÍTULOS

B

CAPÍTULO 2

1. Porque, sem a formatação, não há jeito de localizar nenhuma área específica no disco.
2. O diretório do disco é um índice de nomes, posições e tipos de todos os arquivos no disco.
3. Um arquivo de disco é um bloco individual de informação armazenado no disco, com um nome especificando-o.
4. A informação armazenada na memória é volátil. Ela é destruída ao desligarmos o computador ou ao ser executado o comando NEW, LOAD, DISKINI, BACKUP ou COPY. (Discutiremos BACKUP e COPY no próximo capítulo.) A informação armazenada em disco é permanente. Ela não será destruída se o computador for desligado ou se a memória for apagada (CLEAR). (Não deixe o disco no drive quando você desligar o computador. Explicaremos as razões dessa medida no próximo capítulo.)
5. O único modo de alterar o conteúdo de um arquivo de disco é armazenando outras informações com o mesmo nome de arquivo.

CAPÍTULO 3

1. Porque esse procedimento pode acarretar danos nas informações gravadas em seu disco.
2. Apenas canetas com ponta porosa. Canetas com ponta dura e lápis podem danificar o disco e modificar a informação nele contida.
3. Mensagens de erro dizem a você que alguma coisa está errada no programa que você está executando ou no último comando usado.
4. "Proteção contra gravação" é o modo de proteger seus discos de alterações. Isto é feito colocando um selo sobre o entalhe de proteção contra gravação. Você pode ler um disco protegido contra gravação, mas não pode gravar nele.
5. Em um sistema com um drive, deve-se inserir o disco fonte no drive e digitar BACKUP 0 ENTER. O computador pede a você para inserir o disco destino e pressionar ENTER. Este procedimento se repete até que o computador imprima OK. Em um sistema multidrive, digite o comando BACKUP especificando o número dos drives com o disco fonte e o disco destino. Por exemplo, BACKUP 0 TO 1 copia o disco fonte (no drive 0) no disco destino (no drive 1).

CAPÍTULO 4

1. O nome de um arquivo pode ser alterado com o comando RENAME. Por exemplo, RENAME "ARQANT/NOM" TO ARQNOV/NOM" altera ARQANT/NOM PARA ARQNOV/NOM. Você deve especificar a extensão de ambos para que o computador possa encontrá-los.
2. Você pode descobrir o espaço restante de armazenamento, digitando PRINT FREE(0) ENTER. Ele lhe diz o número de blocos restantes no disco do drive 0. Se você precisar de alguns blocos, elimine (KILL) alguns arquivos ou mude-os para outro disco.
3. A menos que você especifique, o computador sempre usará o drive 0. Isto pode ser modificado, digitando-se DRIVE 1. Dessa forma, você terá acesso ao drive 1, sem ter que especificar no seu comando. (Isto é, agora DIR e DIR1 lhe fornecem o diretório de disco no drive 1.)

CAPÍTULO 5

1. Buffer #1 é uma área de armazenamento temporário da informação que trafega entre o disco e a memória.
2. Um arquivo de disco deve ser aberto (OPEN) antes que qualquer informação possa trafegar entre o disco e a memória.
3. Um arquivo de disco deve ser fechado (CLOSE) para que a informação ainda no buffer termine onde devia e o arquivo possa ser reaberto. Todos os arquivos devem ser fechados antes da troca de discos.
4. Um arquivo aberto (OPEN) para entrada permite que a informação seja transferida do arquivo de disco à memória do computador. Um arquivo aberto (OPEN) para saída permite que a informação seja transferida da memória para o arquivo de disco.
5. Não. Quando você abre (OPEN) um arquivo de acesso seqüencial, você pode abri-lo ou para "I" ou para "O" — nunca para ambos.
6. Não. O arquivo deve primeiro ser fechado, e então reaberto para introdução de dados.

CAPÍTULO 6

1. Registros são divisões do seu arquivo de disco, onde você pode colocar dados. Como cada um tem o mes-

mo tamanho, o computador pode usá-los para ter acesso direto a seus dados.
2. Campos são subdivisões dos registros.
3. No arquivo de acesso seqüencial as únicas posições que o computador conhece são o início e o fim do arquivo. Em um arquivo de acesso direto, você pode determinar onde cada registro individual está (pelo tamanho dos registros).
4. Como cada registro de um arquivo está em uma posição conhecida, o computador pode ter acesso a ele sem ter que passar pelas partes precedentes do arquivo.

CAPÍTULO 7

1. O menor tamanho de um arquivo de disco é 2 304 bytes (um bloco). Como o computador distribui o espaço de disco em blocos, o arquivo não pode ser menor que um bloco.
2. O computador primeiro grava o sinal do número (um sinal de menos se ele for negativo e um espaço em branco se positivo). Depois, ele grava (WRITE) o número e, a seguir, uma seqüência de espaços em branco.
3. Uma string é gravada entre aspas.
4. INPUT introduz apenas os itens de dados listados, enquanto que LINE INPUT introduz tudo até o caractere ENTER.
5. Uma vírgula faz com que o computador desloque para a próxima coluna de impressão antes de imprimir outro item de dados.
6. Um ponto e vírgula faz com que o computador imprima os itens de dados lado a lado.
7. Uma string é impressa sem as aspas.

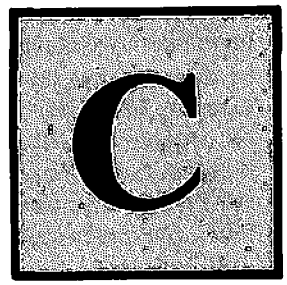
CAPÍTULO 8

1. O computador definirá um registro com 256 bytes de comprimento.
2. O dado deve ter um campo com um comprimento específico para que ele seja justificado à esquerda. Este comprimento é definido na linha FIELD.
3. MKN$ converte um número em uma string codificada de 5 bytes.

CAPÍTULO 9

1. Você deve digitar um A no final do seu comando SAVE se você planeja concatená-lo (MERGE) com um programa na memória.
2. O número da linha do programa do disco prevalecerá.
3. O computador reserva dois buffers.
4. O computador reserva um total de 256 bytes de espaço de buffer quando você o aciona.
5. FILES 3,3000 faz com que o computador reserve 3 buffers com um total de 3000 bytes de espaço.

EXEMPLO DE PROGRAMAS



EXEMPLO DE PROGRAMA #1

Controle seu talão de cheque

Este programa cria um arquivo de disco principal que contém todos seus cheques e depósitos para o ano inteiro. Você pode imprimi-los pelo mês ou pelo ano. Se você quer usar sua impressora, troque as linhas PRINT apropriadas por PRINT # −2. (Ver Capítulo 10 do manual CP 400.)

```
10 'PROGRAMA TALAO DE CHEQUE
20 '
30 'ESTE PROGRAMA FORNECE UM RELATORIO
DE SEUS CHEQUES E SALDOS.
40 'OS CHEQUES PODEM SER IDENTIFICADOS
PELO NUMERO DA CONTA, PARA MOSTRAR O
TIPO DE DESPESA, POR EXEMPLO:
50 'MEDICO, ALUGUEL, ALIMENTACAO ETC.
60 'O PROGRAMA USA ENDERECAMENTO
DIRETO.
70 'CADA REGISTRO DE ARQUIVO POSSUI 45
BYTES, E E' FORMATADO COMO A SEGUIR:
80 '8 BYTES PARA A DATA, 4 BYTES PARA O
NUMERO DO CHEQUE OU O RECIBO DO
DEPOSITO, 20 BYTES PARA O BENEFICIARIO
90 '3 BYTES PARA O NUMERO DA CONTA, E 10
BYTES PARA O VALOR DO CHEQUE OU
DEPOSITO.
100 '
110 CLEAR 1000
120 DIM CHQ$(50)
130 CLS
140 PRINT@107,"SELECAO"
150 PRINT@162,"<1> ACRESCENTA CHEQUES
EM SEU"
155 PRINT@198,"ARQUIVO."
160 PRINT@226,"<2> LISTA CHEQUES,
DEPOSITOS,"
170 PRINT@262,"E SALDOS."
180 PRINT@290,"<3> FIM"
190 PRINT@393,"<1,2 OU 3>"
200 AN$=INKEY$
210 IF AN$="" THEN 200
220 ON VAL(AN$) GOSUB 250,630,1020
230 GOTO 130
240 '
250 ' ESTA SUB-ROTINA INTRODUZ OS DADOS.
260 '
270 OPEN"D",#1,"CHEQUES/DAT",45
280 FIELD#1,8 AS DAT$,4 AS NUM$, 20 AS REC$,
3 AS NCON$, 10 AS AMT$
290 R=LOF(1)
300 R=R+1
310 CLS
320 PRINT@64,"CHEQUE OU DEPOSITO
<C/D>?"
330 AN$=INKEY$
340 IF AN$="D" THEN 370
350 IF AN$="C" THEN 430
360 GOTO330
370 INPUT"DATA DO DEPOSITO (DD/MM/AA)"; D$
380 INPUT"NUMERO DO RECIBO DE
DEPÓSITO(NNNN)";N$
390 P$=""
400 INPUT"NUMERO DA CONTA(NNN)";A$
410 INPUT"VALOR DO DEPOSITO";AMT
420 GOTO490
430 INPUT"DATA DO CHEQUE (DD/MM/AA)";D$
440 INPUT"NUMERO DO CHEQUE (NNNN)";N$
450 INPUT"PAGO A";P$
460 INPUT"NUMERO DA CONTA(NNN)";A$
470 INPUT"VALOR DO CHEQUE";AMT
480 AMT=−AMT
490 LSET DAT$=D$
500 LSET NUM$=N$
510 LSET REC$=P$
520 LSET NCON$=A$
530 LSET AMT$=MKN$(AMT)
540 PUT#1,R
550 PRINT@320,"MAIS ENTRADAS<S/N>"
560 AN$=INKEY$
570 IF AN$="N" THEN 600
580 IF AN$="S" THEN 300
590 GOTO 560
600 CLOSE#1
610 RETURN
620 '
630 'ESTA SUB-ROTINA FAZ O BALANCO DO
SEU TALAO DE CHEQUE E FORNECE O
RESULTADO.
640 '
650 OPEN"D",#1,"CHEQUES/DAT",45
660 FIELD#1,8 AS DAT$, 4 AS NUM$, 20 AS
REC$, 3 AS NCON$, 10 AS AMT$
```

```
670 CLS
680 PRINT@161,"VOCE QUER UMA LISTAGEM
PARA"
685 PRINT@195,"<M> UM MES"
690 PRINT@227,"<A> ANO TODO"
700 INPUT A$
710 IF A$="M" THEN PRINT@289,"QUAL
MES(MM)":INPUT MN$
720 BAL=0
730 FOR R=1 TO LOF(1)
740 GET#1,R
750 BAL=BAL+CVN(AMT$)
760 IF A$="M" AND MID$(DAT$,4,2) <> MN$
THEN 960
770 CLS
780 IF REC$=STRING$(20,32) THEN 860
790 PRINT@64,"DATA DO
CHEQUE:":PRINT@80,DAT$
800 PRINT"NUMERO DO
CHEQUE:":PRINT@114,NUM$
810 PRINT"PAGO A:":PRINT@136,REC$
820 PRINT@160,"NUMERO DA
CONTA":PRINT@177,NCON$
830 PRINT"VALOR DO
CHEQUE:":PRINT@207,USING
"$$##########";-CVN(AMT$)
840 PRINT"SALDO:":PRINT@239,USING"$$##
#########";BAL
850 GOTO 910
860 PRINT:PRINT:PRINT"DATA DO
DEPOSITO:":PRINT@85,DAT$
870 PRINT"NUMERO DO
DEPOSITO:":PRINT@117,NUM$
880 PRINT"NUMERO DA
CONTA:":PRINT@149,NCON$
890 PRINT"VALOR DO
DEPOSITO:":PRINT@179,USING
"$$##########";CVN(AMT$)
900 PRINT"SALDO:":PRINT@206,USING"$$##
#########";BAL
910 PRINT@256,"PRESSIONE <ENTER> PARA
PASSAR PARA O PROXIMO REGISTRO OU <R>
PARA RETORNAR A SELECAO"
920 AN$=INKEY$
930 IF AN$=CHR$(13) THEN 960
940 IF AN$="R" THEN 970
950 GOTO920
960 NEXT R
970 CLOSE#1
980 RETURN
990 '
1000 ' ESTA SUB-ROTINA FINALIZA O
PROGRAMA
1010 '
1020 END
```

EXEMPLO DE PROGRAMA #2

Lista de nomes

Você deseja armazenar os nomes e telefones de seus amigos? Este programa coloca todos eles em um arquivo de disco e em ordem alfabética. Acrescente algumas linhas, e ele armazenará também o endereço e o telefone.

```
NEW
10 ' CRIA UMA LISTA E COLOCA EM ORDEM
ALFABETICA
20 '
30 ' O OBJETIVO DESTE PROGRAMA E' CRIAR
UM ARQUIVO DE NOMES EM ORDEM
ALFABETICA E TELEFONES.
40 ' PRIMEIRAMENTE OS NOMES E NUMEROS
SAO INTRODUZIDOS EM UMA MATRIZ. ARRAY $
(I).
50 ' DEPOIS ORDENADOS ALFABETICAMENTE
E FINALMENTE COLOCADOS EM UM ARQUIVO
60 ' DE DISCO CHAMADO "NOME/NOS" O
ARQUIVO TEM 35 BYTES DE COMPRIMENTO,
TODOS
70 ' ELES ALOCADOS EM UMA VARIAVEL,
INFO$.
80 ' O ARQUIVO PODE SER AUMENTADO A
QUALQUER
90 ' HORA DEPOIS DE SUA CRIACAO. SUA
ORDENACAO E' AUTOMATICA. O PROGRAMA
PODE
100 ' SER USADO EM CONJUNTO COM O
PROGRAMA #3
110 '
120 '
130 '
140 CLEAR 1050
150 DIM ARRAY$(30)
160 OPEN"D",#1,"NOME/NOS",35
170 FIELD#1,35 AS INFO$
180 '
190 ' PRIMEIRO O ARQUIVO E' CHECADO PARA
VER SE ELE CONTEM ALGUMA INFORMACAO.
200 '
210 IF LOF(1)=0 THEN I=1:GOTO 290
220 FOR I=1 TO LOF(1)
230 GET#1,I
240 ARRAY$(I)=INFO$
250 NEXTI
260 '
270 ' OS NOVOS NOMES E NUMEROS SAO
INTRODUZIDOS E ENTAO CONCATENADOS
EM UMA STRING, ARRAY$
280 '
290 CLS
300 PRINT@64
310 INPUT"ULTIMO NOME";U$
```

```
320 INPUT"PRIMEIRO NOME";P$
330 INPUT"INICIAL DO NOME DO MEIO";M$
340 INPUT"CODIGO DA AREA";A$
350 INPUT"NUMERO DO TELEFONE";T$
360 ARRAY$(I)=LEFT$(U$+","+P$+" "+M$+
"                    ",24) + A$ + T$
370 PRINT@288,"MAIS DADOS (S/N)?"
380 AN$=INKEY$
390 IF AN$="S" THEN I=I+1:GOTO 290
400 IF AN$="N" THEN 450
410 GOTO 380
420 '
430 ' DEPOIS ARRAY$(I) E' COLOCADO EM
ORDEM ALFABETICA.
440 '
450 FOR J=1 TO I
460 FOR K=J TO I
470 IF ARRAY$(J)<ARRAY$(K) THEN 510
480 TEMP$=ARRAY$(J)
490 ARRAY$(J)=ARRAY$(K)
500 ARRAY$(K)=TEMP$
510 NEXT K
520 NEXT J
530 '
540 ' FINALMENTE, A LISTA E' TRANSFERIDA
PARA "NOME/NOS".
550 '
560 FOR N=1 TO I
570 LSET INFO$=ARRAY$(N)
580 PUT#1,N
590 NEXT N
600 CLOSE#1
610 END
```

EXEMPLO DE PROGRAMA #3

Procurando um nome

Como o arquivo que você criou no "Exemplo de Programa #2" já está em ordem alfabética, qualquer nome poderá ser facilmente encontrado.
Este programa mostra como:

```
10 ' PROCURAR EM UMA LISTA
20 '
30 ' (NOTA: ESTE PROGRAMA NECESSITA DE
UM ARQUIVO CHAMADO "NOME/NOS".
40 ' VEJA O EXEMPLO DE PROGRAMA #2)
50 ' ESTE PROGRAMA BUSCA UM ARQUIVO DE
DISCO QUE MANTEM NOMES E NUMEROS DE
TELEFONES EM ORDEM ALFABETICA.
60 ' O ARQUIVO E' DE ACESSO
70 ' DIRETO E SE CHAMA "NOME/NOS".
POSSUI 35 BYTES DE COMPRIMENTO, E
80 ' E' FORMATADO COMO A SEGUIR: 24
BYTES PARA O NOME; 3 BYTES
90 ' PARA O CODIGO DA AREA; 8 BYTES PARA
O NUMERO DO TELEFONE.
100 ' O PROGRAMA UTILIZA UM PROCESSO
INTERATIVO DE BUSCA.
110 '
120 OPEN"D",#1,"NOME/NOS",35
130 FIELD#1,24 AS NOME$,3 AS AREA$,8 AS
FONE$
140 CLS
150 PRINT@103,"INTRODUZA NOME:"
155 PRINT@133,"ULTIMO,PRIMEIRO MEIO"
160 LINE INPUT NM$
170 '
180 ' INICIALIZACAO DE VARIAVEIS.
190 '
200 N1$=NM$
210 IF LEN(NM$)<24 THEN 800
220 IF LEN(NM$)>24 THEN 820
230 PRIMEIRO=1
240 MEIO=INT((LOF(1)+1)/2)
250 ULTIMO=LOF(1)
260 CNT=0
270 '
280 ' PRIMEIRO O PROGRAMA CHECA O
ULTIMO REGISTRO,
290 ' PORQUE ELE NAO SERA' CHECADO DE
FORMA REGULAR.
300 '
310 GET#1,ULTIMO
320 IF NOME$=NM$ THEN 450
330 '
340 ' O PROGRAMA COMPARA NM$ COM
NOME$ DO REGISTRO MEIO
350 ' ATE' QUE NM$ SEJA ENCONTRADO OU
QUE JA' TENHA CHECADO REGISTROS
360 ' SUFICIENTES PARA COMPROVAR SUA
INEXISTENCIA.
370 '
380 GET#1,MEIO
390 IF CNT>(LOF(1)+1)/2 THEN 710
400 IF NOME$<NM$ THEN 570
410 IF NOME$>NM$ THEN 640
420 '
430 ' QUANDO NM$ E' ENCONTRADO ELE E'
IMPRESSO.
440 '
450 CLS
460 PRINT@97,NOME$
470 PRINT@136,"(";AREA$;")";FONE$
480 PRINT@193,"PRESSIONE<ENTER>PARA
CONTINUAR,"
490 PRINT@225,"OU PRESSIONE<D>PARA
DESISTIR."
500 AN$=INKEY$
510 IF AN$="D" THEN CLOSE:END
520 IF AN$=CHR$(13) THEN 140
530 GOTO500
540 '
```

```
550 ' SUBPROGRAMA PARA O CASO DE
NOME$<NM$
560 '
570 PRIMEIRO=MEIO
580 MEIO=(MEIO+ULTIMO)/2
590 CNT=CNT+1
600 GOTO380
610 '
620 ' SUBPROGRAMA PARA O CASO DE
NOME$>NM$
630 '
640 ULTIMO=MEIO
650 MEIO=INT((MEIO+PRIMEIRO)/2)
660 CNT=CNT+1
670 GOTO380
680 '
690 ' SUBPROGRAMA PARA O CASO EM QUE
NM$ NAO E' ENCONTRADO.
700 '
710 CLS
720 PRINT@132,N1$;" NAO ENCONTRADO"
730 PRINT@196,"PRESSIONE<ENTER>PARA
TENTAR NOVAMENTE"
740 AN$=INKEY$
750 IF AN$="" THEN 740
760 GOTO 140
770 '
780 ' SUBPROGRAMA QUE CONVERTE NM$
NUMA STRING DE 20 BYTES.
790 '
800 NM$=NM$+" "
810 GOTO210
820 NM$=LEFT$(NM$,24)
830 GOTO 220
```

EXEMPLO DE PROGRAMA #4

Atualização de uma lista

```
10 ' EDITAR SEU ARQUIVO DE NOMES
20 '
30 ' O OBJETIVO DESTE PROGRAMA E' EDITAR
O ARQUIVO "NOME/NOS"
40 ' DO EXEMPLO DE PROGRAMA #2.
50 ' O PROGRAMA PODE MANTER UM
REGISTRO, MUDAR UMA DAS VARIAVEIS NO
REGISTRO,
60 ' OU ENTAO ELIMINAR O REGISTRO INTEIRO
DO ARQUIVO.
70 '
80 CLS
90 PRINT@106,"SELECAO:"
100 PRINT@168,"<1>EDITAR REGISTRO"
110 PRINT@200,"<2>ANULAR REGISTRO"
120 PRINT@232,"<3>FIM"
130 PRINT@298,"<1,2 OU 3>"
140 AN$=INKEY$
150 IF AN$="" THEN 140
160 ON VAL(AN$) GOSUB 180,590,850
170 GOTO 80
180 OPEN"D",#1,"NOME/NOS",35
190 FIELD#1,24 AS NOME$,3 AS AREA$,8 AS
FONE$
200 FOR I=1 TO LOF(1)
210 GET#1,I
220 CLS
230 PRINT@68,"NUMERO DO REGISTRO:";I
240 PRINT@100,"NOME:";NOME$
250 PRINT@132,"CODIGO DA AREA:";AREA$
260 PRINT@164,"NO.DO TELEFONE:";FONE$
270 PRINT@228,"EDITAR ESTE REGISTRO(S/N)"
280 AN$=INKEY$
290 IF AN$="S" THEN 320
300 IF AN$="N" THEN 560
310 GOTO280
320 PRINT@260,"EDITAR NOME?(S/N)"
330 AN$=INKEY$
340 IF AN$="N" THEN NM$=NOME$:GOTO 400
350 IF AN$="S" THEN 370
360 GOTO 330
370 LINE INPUT "    NOVO NOME ";NM$
380 IF LEN(NM$)<24 THEN NM$=NM$+"
":GOTO 380 ELSE 390
390 IF LEN(NM$)>24 THEN NM$=LEFT$(NM$,24)
400 PRINT@292,"EDITAR CODIGO DE
AREA?(S/N)"
410 AN$=INKEY$
420 IF AN$="S" THEN 450
430 IF AN$="N" THEN A$=AREA$:GOTO460
440 GOTO410
450 INPUT "    NOVO COD. DE AREA";A$
460 PRINT@324,"EDITAR NO. DO
TELEFONE?(S/N)"
470 AN$=INKEY$
480 IF AN$="S" THEN 510
490 IF AN$="N" THEN P$=FONE$:GOTO520
500 GOTO 470
510 INPUT"    NOVO NO.DO TELEFONE";T$
520 LSET NOME$=NM$
530 LSET AREA$=A$
540 LSET FONE$=T$
550 PUT#1,I
560 NEXT I
570 CLOSE#1
580 RETURN
590 OPEN"D",#1,"NOME/NOS",35
600 FIELD#1,24 AS NOME$,3 AS AREA$,8 AS
FONE$
610 OPEN"D",#2,"TEMP/FIL",35
620 FIELD#2,24 AS TNOME$, 3 AS TAREA$, 8 AS
TFONE$
630 FOR I=1 TO LOF(1)
640 GET#1,I
650 CLS
660 PRINT@68,"REGISTRO=";I
```

```
670 PRINT@100,"NOME:";NOME$
680 PRINT@132,"CODIGO POSTAL:";AREA$
690 PRINT@164,"NO. DO TELEFONE:";FONE$
700 PRINT@228,"ANULAR ESTE REGISTRO?(S/N)"
710 AN$=INKEY$
720 IF AN$="S" THEN 800
730 IF AN$="N" THEN 750
740 GOTO 710
750 LSET TNOME$=NOME$
760 LSET TAREA$=AREA$
770 LSET TFONE$=FONE$
780 J=J+1
790 PUT#2,J
800 NEXT I
810 CLOSE
820 KILL"NOME/NOS"
830 RENAME "TEMP/FIL" TO "NOME/NOS"
840 RETURN
850 END
```

EXEMPLO DE PROGRAMA #5

Teste de classificação

Este programa é ideal para professores. Ele cria vários arquivos de alunos e suas notas. Você pode encontrar então a média e o desvio padrão da classe inteira ou de cada aluno.

```
10 ' PROGRAMA DE TESTE
20 '
30 ' O OBJETIVO DESTE PROGRAMA E'
INTRODUZIR VARIOS ARQUIVOS
40 ' UM ARQUIVO DE NOMES E VARIOS
ARQUIVOS DE TESTE
50 ' VOCE PODE TER ACESSO AOS ARQUIVOS. E
A
60 ' NOTA DO TESTE PODE SER PROCESSADA
PARA SE ENCONTRAR O DESVIO
70 ' PADRAO. OS ARQUIVOS SAO TODOS DE
ACESSO SEQUENCIAL.
80 '
90 '
100 ' MODULO PRINCIPAL DO PROGRAMA
110 '
120 DIM NOME$(30),GRAU(6,30)
130 CLS
140 PRINT@105,"SELECAO"
150 PRINT@162,"<1>CRIA UM ARQUIVO DE
NOMES"
160 PRINT@194,"<2>CRIA UM ARQUIVO PARA
CADA"
170 PRINT@229,"TESTE"
180 PRINT@258,"<3>PROCESSA NOTA"
190 PRINT@290,"<4>FIM"
200 PRINT@358,"<1, 2, 3 OU 4>"
210 AN$=INKEY$
220 IF AN$="" THEN 210
230 ON VAL(AN$) GOSUB 280,420,630,1420
240 GOTO 130
250 '
260 ' ESTA SUB-ROTINA CONSTROI UM ARQUIVO
"NOMES"
270 '
280 OPEN"O",#1,"NOMES/ALU"
290 CLS
300 PRINT@96,"INTRODUZA O NOME DO
ALUNO: "
310 LINE INPUT NOME$
320 WRITE#1,NOME$
330 PRINT@196,"PRESSIONE<ENTER>PARA
ENTRAR":PRINT@228,"OUTRO NOME":
PRINT@292,"PRESSIONE<P>PARA PARAR"
340 AN$=INKEY$
350 IF AN$="" THEN 340
360 IF AN$<>"P" THEN 290
370 CLOSE#1
380 RETURN
390 '
400 ' ESTA SUB-ROTINA CONSTROI O ARQUIVO
DE TESTE.
410 '
420 CLS
430 PRINT@64
440 INPUT"NO.DO ARQUIVO DO TESTE";TF$
450 IF TF$="" THEN 440
460 TF$="TESTE"+TF$
470 OPEN"I",#1,"NOMES/ALU"
480 OPEN"O",#2,TF$
490 IF EOF(1)=-1 THEN 550
500 INPUT#1,NOME$
510 PRINT"NOME: ";NOME$
520 INPUT"NOTA:";VL
530 WRITE#2,VL
540 GOTO490
550 CLOSE#1,#2
560 RETURN
570 '
580 ' ESTA SUB-ROTINA INTRODUZ O ARQUIVO
"NOMES" E O ARQUIVO DE
590 ' TESTE DESEJADO E ENTAO OS PROCESSA
600 ' BASEADO NA CLASSE OU EM CADA
ALUNO, E
610 ' ENTAO IMPRIME O RESULTADO.
620 '
630 OPEN"I",#1,"NOMES/ALU"
640 IF EOF(1)=-1 THEN 680
650 Y=Y+1
660 INPUT #1,NOME$(Y)
670 GOTO 640
680 YEND=Y
690 CLOSE#1
700 CLS
710 PRINT@96
720 INPUT"QUANTOS TESTES EXISTEM";N
730 FOR X=1 TO N
```

```
740 TF$ = "TESTE" + RIGHT$(STR$(X),1)
750 OPEN"I",#1,TF$
760 FOR Y = 1 TO YEND
770 INPUT#1,GRAU(X,Y)
780 NEXT Y
790 CLOSE#1
800 NEXT X
810 CLS
820 PRINT@130,"TOTAL INDIVIDUAL OU DA
CLASSE"
830 INPUT"TOTAL(I/C) ";AN$
840 IF AN$ = "I" THEN 890
850 IF AN$ = "C" THEN 1140
860 '
870 ' ESTA PARTE PROCESSA AS NOTAS DO
ALUNO.
880 '
890 FOR Y = 1 TO YEND
900 CLS
910 PRINT@105,NOME$(Y)
920 PRINT@137,"NOTA: "
930 TES = 0
940 FOR X = 1 TO N
950 PRINT TAB(10) GRAU(X,Y)
960 TES = TES + GRAU(X,Y)
970 NEXT X
980 MED(Y) = TES/N
990 NUM = 0
1000 FOR X = 1 TO N
1010 NUM = (MED(Y) - GRAU(X,Y))2 + NUM
1020 NEXT X
1030 DP = SQR(NUM/N)
1040 PRINT USING"%        %##.##";"MEDIA:
";MED(Y)
1050 PRINT USING"%                %##.##";
"DESVIO PADRAO: ";DP
1060 PRINT"        PRESSIONE<ENTER>PARA
VER O PROXIMO NOME"
1070 AN$ = INKEY$
1080 IF AN$ = CHR$(13) THEN 1090 ELSE 1070
1090 NEXT Y
1100 CLS
1110 PRINT@105,"FIM DE NOMES"
1120 GOTO1340
1130 '
1140 ' ESTA PARTE PROCESSA AS NOTAS PELO
NUMERO DO
1150 ' TESTE PARA TODA CLASSE.
1160 '
1170 INPUT"QUAL O NO. DO TESTE ";X
1180 CLS
1190 PRINT@2,"DADOS PARA O TESTE DE
NUMERO";X
1200 PRINT"NOME";TAB(25)"NOTA"
1210 TTOT = 0
1220 FOR Y = 1 TO YEND
1230 TTOT = TTOT + GRAU(X,Y)
1240 PRINT TAB(1) NOME$(Y);TAB(25)GRAU(X,Y)
1250 NEXT Y
1260 MED = TTOT/YEND
1270 NUM = 0
1280 FOR Y = 1 TO YEND
1290 NUM = NUM + (MED - GRAU(X,Y))2
1300 NEXT Y
1310 DP = SQR(NUM/(YEND - 1))
1320 PRINT:PRINT USING"%.    %#%%##.
##"; "MEDIA DO TESTE#";X;":";MED
1330 PRINT USING "%                %##.##";
"DESVIO PADRAO:  ";DP
1340 PRINT:PRINT"PRESSIONE<ENTER>PARA
CONTINUAR"
1350 PRINT" O PROCESSAMENTO, E <P>PARA
PARAR"
1360 AN$ = INKEY$
1370 IF AN$ = CHR$(13) THEN 810
1380 IF AN$ = "P" THEN 1390 ELSE 1360
1390 RETURN
1400 '
1410 ' ESTA SUB-ROTINA TERMINA O
PROGRAMA.
1420 '
1430 END
```

EXEMPLO DE PROGRAMA #6

Criar um jogo

```
10 ' CENARIOS
20 '
30 ' ESTE PROGRAMA EXIBIRA' TRES CENARIOS
DE UMA CASA.
40 ' CADA CENARIO E' ARMAZENADO EM DISCO
COMO UM ARQUIVO DE PROGRAMA.
50 '
60 ' O OBJETIVO DESTE PROGRAMA E'
MOSTRAR COMO SE PODE TER ACESSO A
OUTRO PROGRAMA,
70 ' ATRAVES DE UM PROGRAMA PRINCIPAL.
80 ' UM PROGRAMA CHAMA O OUTRO E ASSIM
VOCE VAI PODENDO ANDAR DENTRO DE UMA
CASA.
90 ' SE VOCE QUISER, E' POSSIVEL AMPLIAR
ESTE PROGRAMA AMPLIANDO AS
DEPENDENCIAS DE SUA CASA.
100 ' ANTES DE RODAR ESTE PROGRAMA VOCE
DEVE PRIMEIRO ARMAZENA-LOS NO DISCO
110 ' LOAD"CASA/BAS", R
```

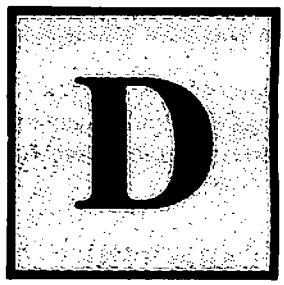
```
10 ' "CASA/BAS"
20 '
30 PMODE3,1
40 PCLS
50 SCREEN 1,0
60 DRAW"BM66,108;D48;R32;U48;L32"
```

```
70 DRAW"BM66,68;R132;BM46,96;R132;BM50,156;
R128"
80 DRAW"BM50,96;D60;BM178,96;D60;BM206,88;
D50"
90 DRAW"BM0,136;R50;BM206,136;R50"
100 LINE(46,96)-(66,68),PSET
110 LINE(178,96)-(198,68),PSET
120 LINE(198,68)-(206,88),PSET
130 LINE(174,156)-(206,136),PSET
140 CIRCLE(92,130)5,0
150 PAINT(0,0),3,4
160 PAINT(0,149),1,4
170 PAINT(67,70),4,4
180 PAINT(55,105),2,4
190 PAINT(194,96),2,4
200 PAINT(82,128),3,4
210 AN$=INKEY$
220 IF AN$="" THEN 210
230 LOAD"SALA/BAS",R
```

```
10 ' "SALA/BAS"
20 '
30 PCLS
40 PMODE3,1
50 SCREEN1,0
60 DRAW"BM104,60;D92;R48;U92;L48"
70 DRAW"BM44,20;R168;D132;L132;BL4;L12;BL4;
L16;BM44,102;U82"
80 DRAW"BM220,60;D100;BM244,58;D126"
90 DRAW"BM42,102;D38;R8;U38;L8"
100 DRAW"BM16,148;D40;R4;U40"
110 DRAW"BM64,148;D40;L4;U40"
120 DRAW"BM80,124;D40;L4;U36"
130 CIRCLE(144,108),4
140 CIRCLE(238,117),4
150 CIRCLE(45,140),15,4,.3,0,.7
160 CIRCLE(45,140),15,4,.3,.95,1
170 CIRCLE(53,136),4
180 LINE(0,192)-(16,176),PSET
190 LINE(20,172)-(44,152),PSET
200 LINE(256,192)-(212,152),PSET
210 PAINT(28,8),3,4
220 LINE(0,0)-(44,20),PSET
230 LINE(256,0)-(212,20),PSET
240 LINE(220,60)-(244,59),PSET
250 LINE(16,148)-(44,124),PSET
260 LINE(16,148)-(64,148),PSET
270 LINE(64,148)-(80,124),PSET
280 LINE-(52,124),PSET
290 PAINT(10,10),3,4
300 PAINT(60,32),3,4
310 PAINT(240,20),3,4
320 PAINT(128,64),2,4
330 PAINT(228,70),2,4
340 PAINT(62,156),4,4
350 PAINT(78,150),4,4
360 PAINT(18,156),4,4
370 PAINT(68,128),1,4
380 PAINT(128,156),2,4
390 PAINT(40,140),4,4
400 PAINT(48,120),2,4
410 CIRCLE(46,98),5,2,2
420 AN$=INKEY$
430 IF AN$="" THEN 420
440 LOAD"ESCADA/BAS",R
```

```
10 ' "ESCADA/BAS"
20 '
30 PCLS
40 PMODE3,1
50 SCREEN1,0
60 DRAW"BM60,20;R140;D120;L40;U32;L4;D52;R4;
U20;BM160,160;L128;U150"
70 DRAW"BM4,62;D130;BM28,166;U102;BM144,148:
R12"
80 DRAW"BM40,72;D24;R36;U24;L36;BM44,76;D16;
R28;U16;L28"
90 LINE(32,12)-(92,12),PSET
100 LINE-(100,20),PSET
110 LINE(0,0)-(60,20),PSET
120 LINE(200,20)-(255,0),PSET
130 LINE(200,140)-(255,192),PSET
140 LINE(0,192)-(32,160),PSET
150 LINE(4,62)-(28,64),PSET
160 PAINT(120,4),2,4
170 PAINT(20,20),2,4
180 PAINT(230,20),2,4
190 PAINT(120,40),2,4
200 PAINT(60,16),3,4
210 PAINT(20,64),3,4
220 PAINT(158,124),4,4
230 PAINT(42,74),4,4
240 LINE(28,8)-(144,148),PSET
250 LINE(64,12)-(156,122),PSET
260 LINE(68,12)-(156,116),PSET
270 DRAW"BM144,148;U38"
280 FOR I=0TO 9
290 DRAW"BM-8,28;U38"
300 NEXT I
310 DRAW"BM56,40;U28;BM48,31;U18;BM40,22;
U10"
320 PAINT(56,84),2,4
330 CIRCLE(56,86),10,3,.4,0,.5
340 LINE(51,86)-(63,86),PSET
350 DRAW"BM56,84;L4;E7;D8"
360 FOR I=1 TO 32
370 CIRCLE (120,176),I*2,I/4,.25
380 NEXT I
382 DRAW"BM232,176;U100;R2;D100"
383 CIRCLE(232,180),15,4,1,.5,0
384 CIRCLE(232,178),6,4,2,.55,.1
385 CIRCLE(232,80),15,4,1,0,.5
386 CIRCLE(232,82),6,4,2.1,.55
390 AN$=INKEY$
400 IF AN$="" THEN 390
```

CÓDIGOS DE CARACTERES



CÓDIGOS DE CARACTERES ASCII

Estes são os códigos ASCII para cada um dos caracteres de seu teclado. A primeira coluna é o caractere; a segundo é o código em notação decimal; e a terceira é o seu equivalente hexadecimal (número na base 16). O Capítulo 8 do manual do CP400, mostra como usar estes códigos com o CHR$ para produzir um caractere.

CARACTERE	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL	CARACTERE	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL
ESPAÇO	32	20	A	65	41
!	33	21	B	66	42
"	34	22	C	67	43
#	35	23	D	68	44
$	36	24	E	69	45
%	37	25	F	70	46
&	38	26	G	71	47
'	39	27	H	72	48
(	40	28	I	73	49
)	41	29	J	74	4A
*	42	2A	K	75	4B
+	43	2B	L	76	4C
,	44	2C	M	77	4D
-	45	2D	N	78	4E
.	46	2E	O	79	4F
/	47	2F	P	80	50
0	48	30	Q	81	51
1	49	31	R	82	52
2	50	32	S	83	53
3	51	33	T	84	54
4	52	34	U	85	55
5	53	35	V	86	56
6	54	36	W	87	57
7	55	37	X	88	58
8	56	38	Y	89	59
9	57	39	Z	90	5A
:	58	3A	↑	94	5E
;	59	3B	↓*	10	0A
<	60	3C	←	8	08
=	61	3D	→*	9	09
>	62	3E	BREAK	03	03
?	63	3F	CLEAR	12	0C
@	64	40	ENTER	13	0D

** Se usado com SHIFT, os códigos destes caracteres são como a seguir: CLEAR é 92 (hexa 5C); ↑ é 95 (hexa 5F); ← é 91 (hexa 5B); ↓ é 21 (hexa 15); e → é 93 (hexa 5D).*

CÓDIGOS DE MINÚSCULAS

Estes são os códigos ASCII das letras minúsculas. Você pode produzir estes caracteres pressionando as teclas SHIFT e ⊡ simultaneamente para obter o modo maiúscula/minúscula. As letras minúsculas aparecem na sua tela com a cor invertida (verde em fundo preto).

CARACTERE	CÓDIGO DECIMAL	CÓDIGO HEXADECIMAL	CARACTERE	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

MAPA DE MEMÓRIA

DECIMAL	HEX	CONTEÚDO	DESCRIÇÃO
0-255	0000-00FF	RAM página direta do sistema	Veja Capítulo 19 do manual CP 400
256-1023	0100-03FF	RAM página ampliada	
1024-1535	0400-05FF	Memória de vídeo	
1536-2440	0600-0988	RAM do sistema adicional	Usado exclusivamente pelo BASIC de Disco
2441 - topo da RAM	0989 - topo da RAM	As posições da memória de Acesso Randômico são alocadas dinamicamente e contêm o seguinte:	
Para sistemas de 16kB o topo da RAM é 16383	Para sistemas de 16kB o topo da RAM é 3FFF	1. Área de buffer de arquivo direto	Espaço total de buffer para arquivos de acesso direto. Na inicialização, são reservados 256 bytes. Este valor é colocado pela instrução FILES.
Para sistemas de 32kB o topo da RAM é 32767	Para sistemas de 32kB o topo da RAM é 7FFF	2. Blocos de controle de arquivo (FCBs)	Dados de controle em cada buffer de usuário. Na inicialização são reservados 843 bytes. Este valor é colocado pela instrução FILES: (número de buffers definidos pela FILES + 1) × 281 bytes.
		3. Memória Gráfica de Vídeo	Espaço reservado para páginas gráficas. 6144 bytes ou 4 páginas são reservadas na inicialização com PCLEAR. Este espaço é calculado pelo número de páginas reservadas por PCLEAR × 1536 bytes por página. (Nota: Todas as páginas devem iniciar numa posição de memória divisível por 256.)
		4. Armazenamento do programa em BASIC 5. Armazenamento da variável BASIC 6. Pilha	Espaço reservado para variáveis e programas em BASIC. 6455* bytes (sistema de 16kB) ou 22839* bytes (sistema de 32kB) são reservados na inicialização. Este valor depende de diferentes ajustes dos buffers de arquivo direto, FCBs, memória gráfica de vídeo, espaço string ou memória do usuário.
		7. Espaço String	Espaço total para dados string. Na inicialização, são reservados 200 bytes, mas isto pode ser modificado pela instrução CLEAR.

		8. Memória do usuário	Espaço total para rotinas de linguagem de máquina do usuário. Não é reservado nenhum espaço para esse fim na inicialização, mas este pode ser alterado pela instrução CLEAR.
32768-49151	8000-BFFF	Interpretador COLOR BASIC	Memória de apenas leitura
49152-57343	C000-DFFF	BASIC DISCO	Memória de apenas leitura
57344-65279	E000-FEFF	Não usado	
65280-65535	FF00-FFFF	Entrada/Saída	

* *Se você executar o comando PRINT MEM, na inicialização, obterá um número um pouco menor que este devido a quantidade de memória gasta para executar este comando.*

INFORMAÇÕES TÉCNICAS

Neste apêndice, discutiremos os detalhes técnicos que envolvem a programação. Essas informações são completamente prescindíveis para aqueles que desejam apenas programar em BASIC. Na verdade, você nem precisaria estar ciente desses detalhes.
Entretanto, se você deseja escrever programas em linguagem de máquina ou simplesmente está interessado em saber tudo que puder a esse respeito, vai querer ter este material. Iniciaremos discutindo como o computador organiza todos os bytes no disco. Depois, mostraremos como ter acesso a eles através de programas em linguagem de máquina e outras técnicas avançadas.

O CONTEÚDO DO DISCO

O computador organiza os bytes no disco em trilhas e setores. Alguns destes bytes controlam o sistema. Mas a maior parte se destina ao armazenamento de dados.

TRILHAS

O computador organiza o disco em 35 trilhas, numeradas de 0 a 34. Cada trilha contém aproximadamente 6250 bytes*. 6084 deles estão divididos em setores; o restante é para os controles do sistema.

Byte #	**Conteúdo**
0-31	Controles do sistema
32-6115	Setores
6116-6249*	Controles do sistema

Todos os bytes de controle do sistema contêm o valor de 4E (hexadecimal).

Nota: Um byte contém 8 bits. Cada bit contém 1 ou 0. Normalmente, expressamos o conteúdo desses bits em números hexadecimais (base 16). Por exemplo, se dissermos que um byte está armazenando o valor 4E hexadecimal, seu conteúdo será 01001110. Você pode encontrar maiores informações sobre sistemas de números hexadecimais e binários em livros de matemática.

*** O número de bytes de controle do sistema no fim de cada trilha pode variar um pouco devido às pequenas variações de velocidade.**

SETORES

Cada trilha contém 18 setores, numerados de 1 a 18. Cada setor compõe-se de 338 bytes sendo que 256 são para dados. Os bytes restantes são para controle do sistema.

Byte #	Conteúdo
0-55	Controles do sistema
56-311	Dados
312-337	Controles do sistema

O conteúdo hexadecimal dos bytes de controle do sistema é:

Byte #	Conteúdo hexadecimal
0-7	00
8-10	F5
11	FE
12	Número da trilha
13	00
14	Número do setor
15	01
16-17	Verificação de Redundância Cíclica (VRC)
18-39	4E
40-51	00
52-54	F5
55	FB
312-313	Verificação de Redundância Cíclica (VRC)
314-337	4E

COMO OS DADOS SÃO ORGANIZADOS

Cada trilha contém 4608 bytes que o computador pode usar para armazenar dados:

```
   18 setores por trilha
× 256 bytes de dados por setor
 4608 bytes de dados por trilha
```

Os bytes de dados na 17ª trilha contêm o diretório do disco. Os bytes de dados das outras 34 trilhas são para arquivos de disco:

Trilha #	Conteúdo dos bytes de dados da trilha
0-16	Arquivos de disco
17	Diretório de disco
18-34	Arquivos de disco

ARQUIVOS DE DISCO

O computador divide as 34 trilhas para arquivos de disco em 68 blocos. Como cada trilha contém 2 blocos, e, portanto, cada bloco, 2304 bytes de comprimento:

```
    9 setores em 1/2 trilha
× 256 bytes de dados por setor
  2304 bytes em um bloco
```

O computador usa blocos para alocar espaço de arquivos de disco em grupos de 2304 bytes. Portanto, se um arquivo contém 4700 bytes, o computador aloca 3 blocos (6912) para esse fim.
A distribuição dos 68 blocos numerados de 0 a 67, é a seguinte:

Trilha 0, Setores 1-9 — Bloco 0
Trilha 0, Setores 10-18 — Bloco 1
Trilha 1, Setores 1-9 — Bloco 2
. .
. .
. .
Trilha 16, Setores 10-18 — Bloco 33
Trilha 17, Setores 1-18 — Diretório
Trilha 18, Setores 1-9 — Bloco 34
. .
. .
. .
Trilha 34, Setores 10-18 — Bloco 67

Nota: O tamanho mínimo de um arquivo de disco é um bloco ou 2304 bytes. Um disco comportará, no máximo, 68 arquivos de disco.

DIRETÓRIO DE DISCO

A trilha do diretório (trilha 17) contém uma tabela de distribuição de arquivos e itens de diretório. Os setores dessa trilha que contêm esta informação são:

Setores #	Conteúdo
2	Tabela de distribuição de arquivo
3-11	Itens do diretório

Os setores restantes na trilha do diretório serão utilizados posteriormente.

ITENS DO DIRETÓRIO

Os 9 setores de diretório (setores 3-11) comportarão até 72 itens. Cada item compõe-se de 32 bytes de comprimento e contém:

Byte #	Conteúdo
0-7	Nome de arquivo, justificado à esquerda, completado com espaços em branco. Se byte 0 = 0, o arquivo foi apagado e o item está livre. Se byte 0 = FF (hexadecimal), o item (e todos os itens seguintes) ainda não foi usado.
8-10	Extensão do nome de arquivo, justificado à esquerda, completado com espaços em branco.
11	Tipo de arquivo 0 = Programa em BASIC 1 = Arquivo de dados em BASIC 2 = Programa em linguagem de máquina 3 = Arquivo-fonte do editor de texto
12	Sinalizador ASCII 0 = O arquivo está em formato binário FF (hexadecimal) = o arquivo está no formato ASCII
13	O número do primeiro bloco do arquivo (0-67).
14-15	O número de bytes em uso no último setor do arquivo.
16-31	Reservado para uso posterior.

TABELA DE DISTRIBUIÇÃO DE ARQUIVOS

O setor 2 do diretório contém uma tabela de distribuição de arquivos para cada um dos 68 blocos no disco. Esta informação está gravada nos primeiros 68 bytes do setor. Os bytes restantes contêm zeros:

Byte #	Conteúdo
0-67	Bloco de informação
68-255	Zeros

Cada um dos primeiros 68 bytes está relacionado com um bloco. Por exemplo, o byte 15 corresponde ao bloco 15. Estes bytes contêm um dos seguintes valores: FF,00-43, ou C0-C9 (hexadecimal).

FF	O bloco correspondente está livre. Ele não faz parte do arquivo de disco.
00-43	O bloco correspondente faz parte do arquivo de disco. O valor convertido em decimal, indica o próximo bloco de arquivo. Por exemplo, se o valor em byte for 0A, 10 será o próximo bloco do arquivo.

C0-C9 O bloco correspondente é o último do arquivo. O valor contido nos bits 0-5 desse byte diz quantos setores daquele bloco fazem parte do arquivo de disco. (Bits 7 e 8 são iguais a 1.)

FATOR DE DEFASAGEM

O computador lê ou grava dados de cada setor individualmente e faz alguns processamentos entre a gravação e leitura de setor.
O disco não pára e espera o computador fazer essas operações. Ele roda continuamente.
Por exemplo, o computador pode ler o setor 1 primeiro. Mas quando o processamento do setor 1 tiver terminado, o disco terá se deslocado para o setor 6.
Para compensar essa diferença, o computador introduz um "fator de defasagem" de 4 quando formata o disco. O computador deve saltar assim 4 setores "físicos" entre cada setor "lógico":

Setor físico	Setor lógico
1	1
2	12
3	5
4	16
5	9
6	2
7	13
8	6
9	17
10	10
11	3
12	14
13	7
14	18
15	11
16	4
17	15
18	8

Portanto, após a leitura do setor 1, o computador saltará os setores físicos 2, 3, 4 e 5. O segundo setor "lógico" que ele lê é o setor "físico" 6. Um fator de defasagem igual a 4 é ideal para se carregar (LOAD) e preservar (SAVE) programas em BASIC. Caso você não trabalhe com a BASIC, poderá usar um fator menor. Por exemplo:

```
DSKINI0, 3
```

diz ao computador para saltar três setores físicos entre cada setor lógico.

Nota: É difícil determinar o fator de defasagem ideal. Recomendamos deixá-lo em 4, a não ser que você conheça muito bem o seu funcionamento.

PROGRAMAÇÃO EM LINGUAGEM DE MÁQUINA DO DISCO

O sistema de disco contém uma rotina de linguagem de máquina chamada DSKCON, com a qual você pode chamar todas as operações de entrada/saída de disco. Para chamar esta rotina, você precisa do conjunto de instruções do microprocessador 6809 do CP 400. Veja o capítulo sobre linguagem de máquina no seu manual para obter maiores informações a esse respeito.

INFORMAÇÃO SOBRE DSKCON

O endereço de entrada do DSKCON é armazenado nas posições C004 e C005 (hexadecimal). Você pode chamá-lo com a seguinte instrução em linguagem Assembly:

```
JSR [$C004]
```

Os parâmetros do DSKCON estão armazenados em seis posições de memória, organizadas dessa forma:

```
DCOPC   RMB   1
DCDRV   RMB   1
DCTRK   RMB   1
DSEC    RMB   1
DCBPT   RMB   2
DCSTA   RMB   1
```

O endereço do primeiro, DCOPC, está contido nas posições C006 e C007 (hexadecimal). Você pode usar as cinco primeiras posições de memória para transferir parâmetros para DSKCON. O DSKCON fornece um byte de status para a sexta posição, DCSTA.
Estes são os parâmetros que você pode transferir para as cinco primeiras posições da memória:

DCOPC — Código de Operação
  0 = Retorna à trilha 0
  1 = Nenhuma operação
  2 = Lê setor
  3 = Grava setor
DCDRV — Número do drive
  0 a 3
DCTRK — Número da trilha
  0 a 34
DSEC — Número do setor
  1 a 18
DCBPT — Indicador de buffer

O endereço de um buffer de 256 bytes. No caso da leitura de um setor, o dado é colocado no buffer. E, no caso da gravação de um setor, os dados no buffer são gravados no disco.

Este é o significado do bytes de status que a rotina DSKCON coloca na posição DCSTA.

DCSTA — Estado

Bit 7 = 1	Disco não preparado
Bit 6 = 1	Gravação protegida
Bit 5 = 1	Gravação defeituosa
Bit 4 = 1	Erro de busca ou Registro não Encontrado
Bit 3 = 1	Erro de VRC
Bit 2 = 1	Dados perdidos

Se todos os bits contiverem zero, nenhum erro terá ocorrido.
Depois de sair de DSKCON, você poderá desligar o motor do drive colocando o valor 0 na posição FF40 (hexa) da memória.

EXEMPLO DE PROGRAMA USANDO DSKCON

Retorno à trilha 0:

```
R LDX    $C006      DEFINE X COMO UM INDICADOR
                    DOS PARAMETROS
  CLR    ,X         DCOPC = 0 PARA RETORNAR A
                    TRILHA 0
  LDA    #$00       DESLIGA O MOTOR DO DRIVE
  STA    $FF40
  LDA    #1         DCDRV = 1 PARA SELECIONAR
                    DRIVE UM
  STA    1,X
  JSR    [$C004]    CHAMA DSKCON
  TST    6,X        VERIFICA ERROS
  BNE    ERRORS     RELATA OS ERROS
  RTS
  LDA    #$45       "E" DE ERRO
  SDA    #41D       CANTO SUPERIOR DIREITO DO
                    VISOR
  RTS
```

Gravação do conteúdo do setor 17 da trilha 3 no drive 0 nas posições de memória de 3800 a 38FF:

```
  LDX    $C006      DEFINE X COMO UM INDICADOR
                    PARA OS PARAMETROS
  LDA    #2         DCOPC = 2 PARA LER UM SETOR
  STA    ,X
  CLR    1,X        SELECIONA DRIVE 0
  LDA    #3         SELECIONA TRILHA 3
```

```
STA     2,X
LDA     #17         SELECIONA SETOR 17
STA     3,X
LDU     #$3800      DCBPT = 3800(HEXA) PARA
                    ARMAZENAR DADOS
STU     4,X
JSR     [$C004]     CHAMA DSKCON
LDA     #$00        DESLIGA O DRIVE DO MOTOR
STA     $FF40
TST     6,X         VERIFICA ERROS
BNE     ERRORS      RELATA OS ERROS
RTS
LDA     #$45        "E" DE ERRO
STA     $41D        CANTO SUPERIOR DIREITO DO
                    VISOR
```

Nota: DSKCON preserva o conteúdo de todos os registros exceto CC.

Você pode gravar um programa similar para gravar um setor, definindo DCOPC como 3 ao invés de 2.

PRESERVANDO UM PROGRAMA EM LINGUAGEM DE MÁQUINA

Você pode usar o comando SAVEM para armazenar um programa em linguagem de máquina no disco. Você precisa especificar onde o programa reside (seu endereço inicial e final) na memória. Você precisa também especificar o endereço onde ele deve ser executado. Use números hexadecimais para todos esses endereços.
Por exemplo, vamos supor que você tenha um programa em linguagem de máquina que reside no endereço 5000-5FFF da memória. O endereço onde ele deve ser executado é 500A. Você armazenaria este programa no disco digitando:

```
SAVEM "PROG/MAQ", &H5000, &H5FFF, &H500A
```

Para carregar novamente na memória, você poderia usar o comando LOADM:

```
LOADM "PROG/MAQ"
```

Este carregaria "PROG/MAQ" na memória nas posições de 5000 a 5FFF. O computador iniciaria sua execução na posição 500A.
Se você deseja carregá-lo em uma posição de memória diferente, pode es-

pecificar um endereço de desvio que será somado aos endereços de carga do programa. Por exemplo:

```
LOADM "PROG/MAQ", 1000
```

carregaria "PROG/MAQ" nas posições de 6000 a 6FFF da memória. O computador iniciaria sua execução no endereço 600A.

COMANDOS ESPECIAIS DE ENTRADA/SAÍDA

A BASIC oferece dois comandos especiais de entrada/saída. Estes comandos introduzem e enviam dados diretamente a um determinado setor. Eles fazem isto desviando (*bypass*) de todo o sistema de arquivo de disco.
O primeiro, DSKI$, introduz os dados do setor que você especifica. Seu formato é o seguinte:

DSKI$ *número do drive, trilha, setor, variável de string 1, variável de string 2*

Os primeiros 128 bytes do setor são introduzidos na *variável de string 1*. Os 128 bytes seguintes são introduzidos na *variável de string 2*. Por exemplo: DSKI$ 0, 17, 1, A$, B$ ENTER lê o conteúdo do setor 1, da trilha 17 do disco no drive 0: os primeiros 128 bytes no A$, e os 128 bytes seguintes no B$. Depois de digitar este comando, você pode exibir o conteúdo deste setor com:

```
PRINT A$; B$ ENTER
```

Como DSKI$ lê qualquer setor no disco, ele é o único comando BASIC que lê o setor do diretório. Este exemplo de programa usa DSKI$ para buscar nomes de arquivo com a extensão "DAT" no diretório:

```
10 FOR X = 3 TO 11
20 DSKI$ 0,17,X, A$,B$
30 C$ = A$ + LEFT$(B$,127)
40 NAM$(0) = LEFT$(C$,8)
50 EXT$(0) = MID$(C$,9,3)
60 FOR N=1 TO 7
70 NAM$(N) = MID$(C$,N*32+1,8)
80 EXT$(N) = MID$(C$,9+N*32,3)
90 NEXT N
100 FOR N = 0 TO 7
110 IF EXT$(N) = "DAT" AND LEFT$(NAM$(N),1)<>CHR$(0)
    THEN PRINT NAM$(N)
120 NEXT N
130 NEXT X
```

O segundo comando, DSKO$, envia dados diretamente para o setor que você especifica. Como ele desvia do sistema de arquivo de disco, ele envia dados sem abrir um arquivo e listar sua posição no diretório. Por esta razão, você precisa ser cuidadoso:

1. Não envie dados para os setores do diretório a menos que você esteja pensando em não utilizar mais o diretório.

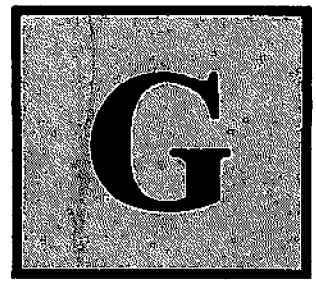
2. Não envie dados sobre outros dados que você armazenou no disco.

O formato de DSKO$ é:

DSKO$ *número de drive, trilha, setor, string 1, string 2*
String 1 vai nos primeiros 128 bytes do setor.
String 2 vai nos próximos 128 bytes. Por exemplo:

DSKO$ 0, 1, 3, "PRIMEIRA STRING", "SEGUNDA STRING" ENTER

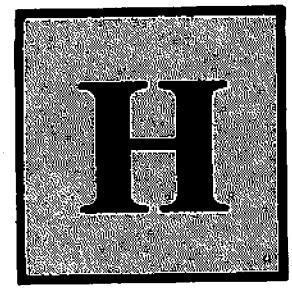
Envia dados ao setor 3 da trilha 1, no disco do drive 0. A "PRIMEIRA STRING" vai nos primeiros 128 bytes deste setor. A "SEGUNDA STRING" vai nos próximos 128 bytes.



ESPECIFICAÇÕES

TIPO DE DISCOS	Disquetes de 5 1/4"
ORGANIZAÇÃO DE DISCO (DISCO FORMATADO)	Face simples Dupla densidade 35 trilhas 18 setores por trilha 256 bytes de dados por setor Diretório na trilha 17
TEMPERATURA DE OPERAÇÃO	18 a 113 graus Fahrenheit 10 a 45 graus centígrados
CAPACIDADE DE MEMÓRIA	218,8 kilobytes por disco
NÃO FORMATADA	6,2 kilobytes por trilha
SETOR	179,1 kilobytes por disco
(setor I/O/trilha)	5,1 kilobytes por trilha
VELOCIDADE DE TRANSMISSÃO DE DADOS	250 kilobits por segundo
TEMPO DE ACESSO	
TRILHA A TRILHA	30 ms
MÉDIA	463 ms
TEMPO DE ACOMODAÇÃO	10 ms
NÚMERO DE ÍNDICES	1
PESO DO DRIVE	5,0 kg

MENSAGENS DE ERRO



/0 Divisão por zero. Pediu-se ao computador para dividir por zero. Você também pode obter esta mensagem de erro se não colocar o nome do arquivo entre aspas.

AE Arquivo já existe. Você está tentando alterar o nome ou copiar um arquivo já existente.

AO Tentativa em abrir um arquivo de dados já aberto.

BR Número de registro defeituoso. Você usou um número de registro ilegal (menor que 1 ou maior que o número máximo de registros do disco) em uma linha PUT ou GET. Use um número de registro diferente na linha PUT ou GET, ou defina um comprimento maior para o registro na linha OPEN.

BS Índice ilegal. Os índices em uma matriz estão fora da faixa. Use DIM para dimensionar a matriz. Por exemplo, se você tiver A(11) em seu programa, sem uma linha DIM precedente que dimensiona a matriz A para 11 ou mais elementos, você obterá esse erro.

CN Impossibilidade de continuação. Se você usar o comando CONT e o programa já tiver acabado, ocorrerá esse erro.

DD Tentativa de redimensionamento de uma matriz. Uma matriz só pode ser dimensionada uma vez. Por exemplo, você não pode ter DIM A(11) e DIM A(20) no mesmo programa.

DF Disco cheio. O disco em que você está tentando armazenar seu arquivo está completo. Use outro disco.

DN Erro de número de drive ou de número de dispositivo.
Erro de número de drive. Você está usando um número de drive maior que 3. Você também obterá este erro se não especificar o número de drive quando usar DSKINI ou BACKUP. Se você tem apenas um drive, especifique o drive 0 com estes comandos (DSKINI0 ou BACKUP0).
Erro de número de dispositivo. Você está usando mais buffers que o computador tem reservado. Use FILES para reservar mais. Você também obterá este erro se usar um número de buffer inexistente (tal como buffer # – 3) ou omitir o buffer (tal como FIELD 1 AS A$ ao invés de FIELD #1, 1 AS A$).

DS Instrução direta. Há uma instrução direta no arquivo de dados. Isto pode acontecer se você carregar um programa sem número de linha.

ER Gravar ou ler dados depois do fim do registro (apenas acesso direto). Você está tentando colocar (PUT ou INPUT) mais dados nos registros do que eles podem conter.

FC Chamada ilegal de função. Isto acontece quando você usa um parâmetro (número) com uma palavra BASIC que está fora da faixa. Por exemplo, SOUND (260,260) ou CLS(10) causará este erro. RIGHT$(S$,20), quando há apenas 10 caracteres em S$, poderá também causar esse erro. Outro exemplo é um índice negativo, tal como A(– 1), ou uma chamada USR antes do endereço ter sido introduzido (POKE).

FD Dados defeituosos em arquivo. Este erro ocorre quando você imprime (PRINT) dados em um arquivo, ou introduz (INPUT) dados de um arquivo, usando um tipo errado de variável para o dado correspondente. Por exemplo, INPUT #1, A, quando os dados no arquivo são string, causa este erro.

FM Modo defeituoso de arquivo. Você especificou o modo de arquivo errado ("O", "I" ou "D") na sua linha OPEN para o que você está tentando fazer. Por exemplo, você pode estar tentando obter (GET) um registro de um arquivo aberto para "I" (use "D"), ou gravar (WRITE) dados em um arquivo aberto para "I" (use "O").

FN Nome defeituoso de arquivo. Você usou um formato inaceitável para dar nome a seu arquivo.

FO Estouro de campo. O comprimento do campo é maior que o comprimento do registro.

FS Estrutura defeituosa de arquivo. Há alguns erros em seu arquivo de disco. Ou os dados foram gravados incorretamente ou a trilha do diretório do disco está defeituosa. Veja as instruções de como proceder neste caso em IO.

ID Instrução direta ilegal. Você pode usar apenas INPUT como uma linha de programa, não como uma linha de comando.

IE Introdução de dados depois do fim do arquivo. Usar EOF ou LOF para verificar se você alcançou o fim do arquivo. Se sim, fechá-lo (CLOSE).

IO Erro de entrada/saída. O computador está tendo problemas para introduzir ou enviar informações para o disco.
(1) Certifique-se de que o disco está inserido adequadamente no drive indicado e que a porta do drive está fechada.
(2) Se este erro ainda persiste, deve haver alguma coisa errada com seu disco. Tente reinserir o primeiro disco. Depois, tente usar um diferente ou reformatá-lo. (Lembre-se de que reformatando um disco, seu conteúdo é apagado.)
(3) Se o erro ainda persiste, você provavelmente tem um problema com o sistema do computador. Chame a assistência técnica. Este erro poderia também ser causado por problemas de interface com outro dispositivo, tal como um gravador de fita.

LS String muito longa. Uma string pode conter apenas 255 caracteres.

NE O computador não pode encontrar o arquivo de disco que você quer. Verifique o diretório do disco para ver se o arquivo está lá. Se você tiver mais que um drive de disco, pode ser que você não tenha incluído o número de drive apropriado no nome do arquivo. Se você está usando COPY, KILL ou RENAME (discutido no próximo apêndice), você pode ter esquecido da extensão.

NF NEXT sem FOR. NEXT está sendo usado sem seu parceiro, a palavra FOR. Este erro também ocorre quando você tem as linhas NEXT invertidas em um ciclo alojado.

NO Arquivo não aberto. Você não pode receber nem enviar dados para um arquivo até que você o tenha aberto.

OB Insuficiência de espaço para buffer. Use FILES para reservar mais espaço.

OD Insuficiência de dados. Um READ foi executado sem dados suficientes. A instrução DATA pode ter sido omitida do programa.

OM Insuficiência de memória. Toda memória disponível foi usada ou reservada.

OS Insuficiência de espaço string. Não há espaço suficiente na memória para fazer suas operações com string. Use CLEAR no início do seu programa para reservar mais espaço string.

OV Estouro. O número é muito grande para o computador manipular.

RG RETURN sem GOSUB. A linha RETURN foi encontrada sem uma linha GOSUB correspondente.

SE A string não faz parte do buffer. O campo no qual você está tentando justificar os dados à direita ou à esquerda (LSET ou RSET) ainda não foi definido. Verifique a linha FIELD.

SN Erro de sintaxe. Isto pode resultar de um erro de digitação, pontuação incorreta, parênteses aberto ou um caractere ilegal. Redigite a linha de programa ou comando.

ST Fórmula muito complexa de string. A operação de string é muito complexa para ser efetuada. Redefina a operação em etapas menores.

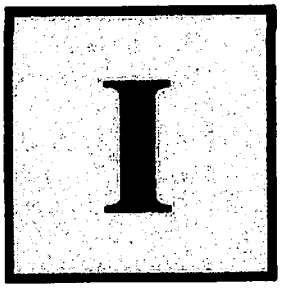
TM Introdução ilegal. Isto ocorre quando você tenta atribuir dados numéricos a uma variável string (A$ = 2) ou dados de string a uma variável numérica (A = "CASA"). Este erro pode também ocorrer se o nome de arquivo não for colocado entre aspas.

UL Linha indefinida. Você tem um GOTO, GOSUB ou outra linha pedindo ao computador para ir para um número de linha inexistente.

VF Verificação. Esse erro só aparece quando o comando VERIFY for ativado e você estiver gravando no disco. O computador está informando que houve uma falha na gravação. Veja IO para aprender o que fazer.

WP Proteção de gravação. Você está tentando armazenar informação em um disco que está protegido contra gravação. Retire o selo de proteção ou use outro disco. Se seu disco não está protegido contra gravação, há algum problema de entrada/saída. Veja IO para se informar sobre o procedimento correto neste caso.

SUMÁRIO DO *BASIC* DE DISCO



Este é um resumo de cada "comando" do BASIC DISCO. Você pode usar também qualquer um dos comandos do COLOR BASIC. (Consulte o seu manual ou o cartão de referência.)
A primeira linha consiste no formato em que se deve digitar o comando. As palavras grifadas representam os "parâmetros" — valores que você pode especificar com os comandos.
Eis o significado de alguns dos parâmetros que você pode especificar:

nome de arquivo
Toda informação armazenada em um disco deve ter um *nome de arquivo*. O *nome de arquivo* deve seguir esse formato:
nome/extensão: número do drive
O *nome* é obrigatório. Deve conter de 1 a 8 caracteres.
A *extensão* é opcional. Pode conter de 1 a 3 caracteres.
O *número do drive* é opcional. Se você não usá-lo quando abrir um arquivo de disco, o computador trabalhará com o drive 0 (ou o drive especificado no comando DRIVE).

número
Pode ser um número (4,3.1), uma variável numérica (A, B, C), uma função numérica (ABS(3)), ou uma operação numérica (5+9,A+6).

string
Pode consistir em caracteres ("G","STRING"), numa variável string (D$,T$), numa função string (LEFT$(S$,7)), ou numa operação string ("D"+U$).

dado
Pode ser um número ou uma string.

COMANDOS E INSTRUÇÕES BASIC

BACKUP *drive fonte* TO *drive destino*
Reproduz o conteúdo do disco do *drive fonte* no disco do *drive destino*. Se você tiver só um drive, especifique-o como *drive fonte*. O computador avisa quando o BACKUP termina. A execução deste comando limpa a memória.

```
BACKUP 0 TO 1      BACKUP 0
```

CLOSE #*buffer*,...
Inibe a comunicação com o *buffer* especificado. (Com referências a números de *buffer* veja OPEN.) Se você omitir o *buffer*, o computador fechará todos os arquivos que estiverem abertos.

```
CLOSE #1      CLOSE #1,#2
```

COPY *nome de arquivo1* TO *nome de arquivo2*
Copia o conteúdo do *nome de arquivo1* em *nome de arquivo2*. Cada *nome de arquivo* deve incluir uma *extensão*. (Veja o formato do *nome de arquivos* a seguir.) A execução deste comando apaga a memória.

```
COPY "ESC/BAS" TO NOVESC/BAS"
COPY "ORC/DAT:0" TO "ORC/DAT:1"
```

CVN (*variável string*)
Converte uma string de 5 bytes (criada por MKN$) novamente no número que ela representa.

```
X = CVN(A$)
```

DIR *número do drive*
Exibe o diretório do disco inserido no drive cujo número você especifica. O computador trabalhará com o drive 0, se for omitido o número do drive. (A menos que você use o comando DRIVE para redefinir esse caso de omissão.) Um exemplo típico do que um diretório exibe é apresentado abaixo:

```
MEUPROG     BAS      0 B 3
SEUPROG     BAS      0 A 1
NOSDADO     DADO     1 A 5
DELPROG     BIN      2 B 2
```

A primeira coluna é o nome do arquivo. A segunda sua extensão. A terceira é o tipo do arquivo (0 = programa BASIC, 1 = arquivo de dados do BASIC, 2 = arquivo em linguagem de máquina, 3 = arquivo fonte editor). A quarta coluna é o formato de armazenamento (A = ASCII, B = Binário). A quinta coluna é o comprimento do arquivo em blocos.

```
DIR0      DIR
```

DRIVE *número do drive*
Altera o número do drive de omissão por aquele que você especifica. Se você não usar o comando DRIVE, o computador trabalhará com o drive 0.

```
DRIVE 1
```

DSKINI *número do drive*
Formata um disco no número de *drive* que você especifica. A execução deste comando limpará a memória.

```
DSKINI0      DSKINI1
```

DSKI$ *número do drive, trilha, setor, variável string1, variável string2*
Lê dados de um certo *setor*, dentro de uma certa *trilha* em um disco inserido no drive especificado pelo número do drive. Os primeiros 128 bytes de dados são introduzidos na *variável string1*; os outros 128 na *variável string2*.

```
DSKI$ 0, 12, 3, M$, N$
```

DSKO$ *número do drive, trilha, setor, string1, string2*
Envia dados string a um *setor*, de uma *trilha* do disco inserido no drive especificado pelo *número do drive*. A *string1* é enviada nos primeiros 128 bytes do setor; *string2*, aos outros 128 bytes. Este comando causará danos ao conteúdo do disco quando usado inapropriadamente.

```
DSKO$ 0, 2, 1, "PRIMEIRO DADO", "SEGUNDO DADO"
```

EOF (*buffer*)
Resulta em zero se há mais dados a serem lidos no *buffer* e em −1 se não há mais dados nele. (Com referência a número de buffer, veja OPEN.)

```
IF EOF(1) = -1 THEN CLOSE #1
```

FIELD # *buffer, tamanho do campo* AS *nome do campo,...*
Divide em campos os espaços de um *buffer* de acesso direto. (Com referência a números de *buffer*, veja OPEN.) Você especifica o *tamanho* e o *nome* de cada *campo*.

```
FIELD #1, 10 AS A$, 15 AS B$, 8 AS D$
```

FILES *número de buffer, tamanho de buffer*
Diz ao computador quantos buffers devem ser reservados na memória (*número de*

buffer) e o total de bytes reservado para esses buffers (*tamanho do buffer*). Se você não usar FILES, o computador reservará espaço suficiente de memória para os buffers 1 e 2, e um total de 256 bytes para cada um.

```
FILES 1, 1000      FILES 5
```

FREE (*número de drive*)
Fornece o número de blocos livres no disco do drive de número especificado.

```
PRINT FREE(0)
```

GET #*buffer*, *número do registro*
Pega o próximo registro ou o registro com o *número* que você especifica, e o coloca no *buffer*. (Com referência a números de buffer, veja OPEN.)

```
GET #1,5  GET #2, 3
```

INPUT #*buffer*, *nome de variável*,...
Introduz dados do *buffer* especificado e atribui um nome de variável a cada item de dados do *buffer*. (Com referência a números de buffer, veja OPEN.)

```
INPUT #1, A$, B$
```

KILL *nome de arquivo*
Elimina o *nome do arquivo* especificado no diretório de disco. (Veja o formato dos nomes de arquivo a seguir.) Você deve incluir a *extensão* no *nome do arquivo*.

```
KILL "ARQ/BAS"      "KILL ARQ/DAT:1"
```

LINE INPUT #*buffer*, *dado*
Introduz uma linha (todos os dados até o caractere ENTER) do *buffer* especificado. (Com referência aos números do *buffer*, veja OPEN.)

```
LINE INPUT #1, X$
```

LOAD *nome de arquivo*, R
Carrega na memória o arquivo de programa BASIC contido num disco. A opção R faz com que o computador rode o programa logo após a carga. Se seu *nome de arquivo* não tiver uma *extensão*, o computador a considerará BAS. (Veja o formato dos *nomes de arquivo*.) A execução deste programa limpa a memória.

```
LOAD "PROGRAMA", R      LOAD "CONTA/BAS:1"
```

LOADM *nome de arquivo*, *endereço de desvio*
Carrega na memória um arquivo em linguagem de máquina contido num disco. Você pode especificar um *endereço de desvio* para ser somado ao *endereço de carga* do programa. Se seu *nome de arquivo* não tiver uma *extensão*, o computador a considerará BIN. (Veja o formato para *nomes de arquivo*.)

```
LOADM "PROG/BIN, 3522
```

LOC(*buffer*)
Fornece o número atual do registro do buffer especificado. (Com referência aos números de *buffer*, veja OPEN.)

```
PRINT LOC(1)
```

LOF(*buffer*)
Fornece o maior número de registro do buffer especificado. (Com referência aos números de *buffers*, veja OPEN.)

```
FOR R = 1 TO LOF(1)
```

LSET *nome do campo* = *dados*
Justifica os dados à esquerda dentro do campo especificado. Se a informação for mais comprida que o campo, os caracteres à direita serão truncados.

```
LSET A$ = "BANANAS"      LSET B$ = T$
```

MERGE *nome de arquivo*, R

Carrega um *arquivo de programa* do disco e o funde com o programa existente na memória. A opção R faz com que o computador rode o programa logo após a fusão. (Veja o formato para *nome de arquivo*.) O arquivo de programa de disco não poderá ser fundido (MERGE) a menos que ele tenha sido preservado (SAVE) com a opção A (ASCII).

```
MERGE "SUB/BAS"      MERGE "NOVO", R
```

MKN$(*número*)

Converte um *número* em uma string de 5 bytes, para armazenamento em um arquivo de disco formatado.

```
LSET B$ = MKN$(460376)
```

OPEN "*modo*", #*buffer*, *nome de arquivo*, *comprimento do registro*

Abre um lugar na memória chamado *buffer* que transfere dados de/para um certo dispositivo. O *buffer* e os dispositivos com os quais eles se comunicam são:

0 — tela ou impressora (não é necessário abrir este buffer)
−1 — gravador de fita
−2 — impressora
1-15 — drive de disco

Os modos de comunicação que você pode usar são:

I — Introdução de dados de um arquivo de acesso seqüencial.
O — Envio de dados para um arquivo de acesso seqüencial.
D — Introdução ou envio de dados para um arquivo de acesso direto.

O *nome de arquivo* usado deve seguir o formato descrito anteriormente. Caso não seja especificada a *extensão*, o computador a considerará como sendo DAT. Se você estiver liberando a comunicação com um arquivo de acesso direto, poderá também especificar o *comprimento do registro*. No caso de omissão ele será de 256 bytes.

```
OPEN "D",#1, "CASA", 19
OPEN "I", #2 "TROC/DAT"
```

PRINT #*buffer*, *lista de dados*

Imprime os *dados* no *buffer*. (Com referência aos números de *buffer*, veja OPEN.) Você pode usar uma vírgula ou um ponto e vírgula para separar cada item na lista de dados.

```
PRINT #1, "BANCO"
```

PRINT #buffer, USING *formato*; *lista de dados*

Imprime dados no *buffer* usando o *formato* que você especifica. O *formato* é uma string que pode especificar um *formato numérico* ou *string*.

Os *formatos numéricos* podem consistir de qualquer um dos seguintes símbolos:

#	formata números
.	formata um ponto decimal
,	coloca uma vírgula à direita de cada três números
**	preenche os espaços dianteiros com asteriscos
$	coloca $ na frente do número
$$	sinal $ flutuante
+	na primeira posição faz com que o sinal seja impresso antes do número; na última posição, com que o sinal seja impresso depois do número
↑↑↑↑	imprime número em notação exponencial
−	imprime um sinal menos depois de números negativos

```
PRINT USING #1, "##.#"; 76.98
PRINT USING #2, "**$#.## −"; −4.786
```

Os *formatos string* podem consistir de:

% % define o comprimento de uma string
! imprime o primeiro caractere da string

```
PRINT USING #1, "!"; "AMARELO"
PRINT USING #1, "% %"; "CANCAO"
```

Consulte o seu manual do CP 400 para maiores informações.

PUT *#buffer, número de registro*
Determina um *número de registro* para os dados no *buffer*. Se você não especifica um *número de registro*, o computador o considera como sendo o registro atual. (Com referência a números de *buffer*, veja OPEN.)

```
PUT #2, 3      PUT #1, 4
```

RENAME *nome antigo de arquivo* TO *nome novo de arquivo*
Altera o nome de um arquivo no disco. Você deve especificar a extensão dos dois *nomes de arquivos*.

```
RENAME "AARQ/DAT:1" TO "BARQ/DAT:1"
```

RSET *nome do campo = dados*
Justifica os dados à direita dentro do campo especificado. Se a informação é maior que o campo, os caracteres à esquerda serão truncados (semelhante a LSET).

```
RSET M$ = "NUMERO"
```

RUN *nome dé arquivo*, R
Carrega o nome de arquivo contido num disco e o executa. A opção R faz com que todos os arquivos abertos permaneçam abertos. (Veja o formato de nomes de arquivos.)

```
RUN "AZUL"      RUN "PROG/BAS", R
```

SAVE *nome de arquivo*, A
Preserva no disco o arquivo especificado. Caso não seja especificada uma extensão, o computador a considerará como sendo BAS. A opção faz com que seu programa seja preservado no formato ASCII. (Veja o formato de *nomes de arquivos*.)

```
SAVE "PROG/BAS"      SAVE "TESTE:1", A
```

SAVEM *nome de arquivo, primeiro endereço, último endereço, endereço de execução*
Preserva o *nome de arquivo* — um programa em linguagem de máquina que inicia no *primeiro endereço* (na memória) e termina no *último endereço*. Você especifica também o endereço em que ele será *executado*. Caso não seja especificada a extensão, o computador a considerará como sendo BIN. (Veja o formato de nomes de arquivos.)

```
SAVEM "ARQ/BIN:1", &H5200, &H5800, &H5300
```

UNLOAD *número de drive*
Fecha qualquer arquivo aberto no *drive* de número especificado. Se você não especificar um *número de drive*, o computador usará o drive 0.

```
UNLOAD 0      UNLOAD
```

VERIFY ON
VERIFY OFF
Liga ou desliga a função de verificação. Quando VERIFY está ligado, o computador checa todas as gravações do disco.

WRITE *#buffer, lista de dados*
Grava os dados no *buffer* especificado. (Com referência aos números de *buffer*, veja OPEN.) Use uma vírgula para separar cada item da lista de *dados*.

```
WRITE #1, A$, B$, C
```

ÍNDICE REMISSIVO