PEEK, POKE & CALL

CLAUDIO MACHADO



Mais de 100 PEEK, POKE, CALL e rotinas úteis para APPLE II+, APPLE IIe e compatíveis.

Editora Aleph

450

CLAUDIO MACHADO

Peek, Poke & Call

ALEXANDRE FERNANDES
Departamento Tecnico

Editora Aleph

CLAUDIO MACHADO

Peek, Poke & Call

1ª EDIÇÃO
1987

Editora Aleph

© 1987 - EDITORA ALEPH

EXPEDIENTE

Coordenação Editorial: Pierluigi Piazzi
Coordenação Pedagógica: Betty Fromer Piazzi
Editoração: Elaine De Carlos
Eduardo Gama Schaeffer
Revisão de Português: José Augusto Queiroz
Revisão Técnica: Oren Prinsky
Ilustrações e Capa: Fernando Gastão Moretti
Arte: Ana Lúcia Antico
Produção Editorial: Rosa Kogan Fromer

ALEPH Publicações e Assessoria Pedagógica Ltda
C.P. 20707 CEP 01498 S.Paulo SP (011) 813-2033

Dados do Catalogação na Publicação (CIP) Internacional
(Câmara Brasileira do Livro, SP, Brasil)

Machado, Cláudio, 1960-
MI3p PEEK, POKE, CALL para Apple : para computadores APPLE II Plus, APPLE IIe e compatíveis / Cláudio Machado. -- São Paulo : Aleph, 1987.
(Segredos do Apple)

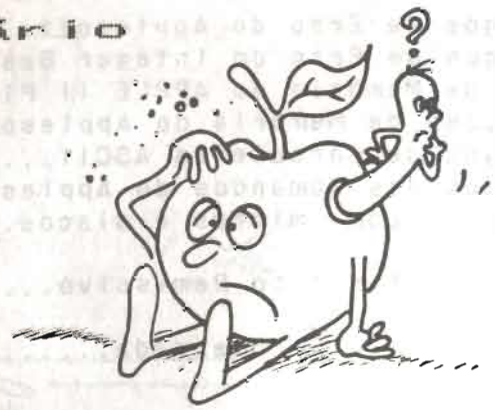
1. Apple II Plus (Computador) - Programação
2. Apple IIe (Computador) - Programação I. Título. II. Série.

87-1101 CDD-001.642

Índices para catálago sistemático:

1. Apple II Plus : Computadores eletrônicos : Programação : Processamento de dados 001.642
2. Apple IIe : Computadores eletrônicos : Programação : Processamento de dados 001.642

Sumário



I. Introdução..... 07
I.1. A quem se destina..... 09
I.2. Definições..... 10
I.3. POKE..... 10
I.4. PEEK..... 10
I.5. CALL..... 11

II. A Função CALL..... 13

III. A Função PEEK..... 26

IV. A Função POKE..... 40

V. Sub-rotinas..... 65
V.1. Colocação de marca personalizada.. 65
V.2. Gerando sons..... 66
V.3. Tabela de instruções do Applesoft. 67
V.4. Tabela de comandos do D.O.S. 3.3.. 68
V.5. Tabela de mensagens de erro..... 70
V.6. Tabela do catálogo..... 71
V.7. Rotina Hidden..... 72
V.8. Relação Caracter X Código ASCII... 73
V.9. Programa conversão Hexa<=>Decimal. 73
V.10. Applesoft X Arquivo Texto..... 76
V.11. Localizador de Endereço Binário... 77

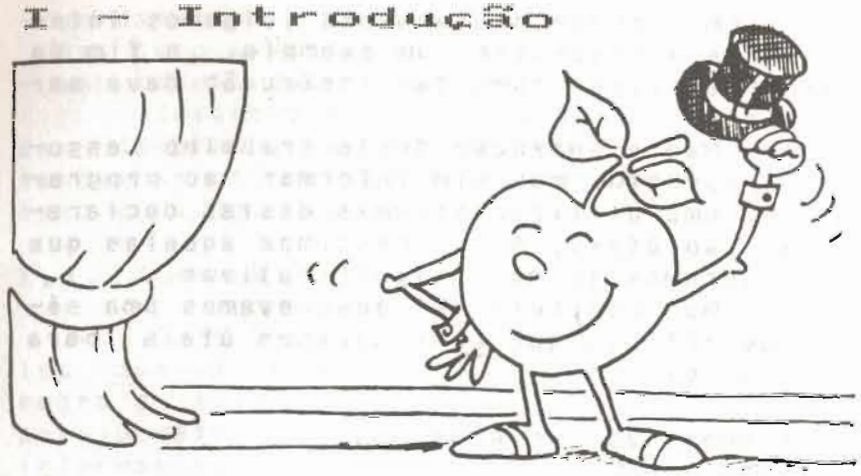
Apêndices

A. Códigos de Erro do Applesoft..... 79
B. Códigos de Erro do Integer Basic.. 87
C. Mapa de Memória do APPLE II Plus.. 89
D. Posições de Memória do Applesoft.. 91
E. Códigos de Caracteres ASCII..... 92
F. Códigos dos Comandos do Applesoft. 94
G. Cuidados com micros e discos..... 98

Indice Alfabético Remissivo..... 100

Bibliografia Recomendada..... 104

I - Introdução



Este livro pretende preencher uma lacuna muito importante dentro do aprendizado da manipulação dos computadores compatíveis com a linha Apple II Plus e Apple IIe.

A maioria dos livros sobre Linguagem Basic ou Assembler faz referência aos PEEKs, POKEs e CALLs de maneira muito superficial. Por isso, reunimos neste trabalho uma listagem das principais declarações da ROM do Apple.

O Apple II Plus pode dispor de 65.536 endereços de memória para armazenar programas e dados. Esses endereços podem referir-se tanto às memorias RAM ou ROM, quanto à possibilidade de comunicação do computador com o mundo exterior. Exemplos dessa comunicação incluem os paddles para jogos, armazenamento de dados em discos, etc.

Os comandos PEEK, POKE e CALL estão descritos neste livro em ordem numérica, facilitando, assim, uma pesquisa rápida.

Em algumas declarações julgamos interessante acrescentar um exemplo, a fim de melhor explicar como tal instrução deve ser utilizada.

Não é intenção deste trabalho esgotar o assunto, mas sim informar ao programador como utilizar algumas destas declarações tão úteis. Aqui reunimos aquelas que consideramos as mais significativas.

No capítulo 'V' descrevemos uma série de rotinas que consideramos úteis para o usuário:

- personalizar um programa;
- gerar sons;
- tabela de instruções do Applesoft;
- tabela de comandos do D.O.S. 3.3;
- tabela das mensagens de erro;
- tabela do catálogo;
- demonstração dos caracteres "hidden";
- relação caracter X código ASCII;
- conversão de números decimais para hexadecimais e vice-versa;
- localizador de endereços binários.

Anexamos, também, a listagem de um programa que tem como objetivo converter valores hexadecimais para decimais e vice-versa.

Ao final deste trabalho, o usuário encontrará os seguintes apêndices:

- uma tabela de erros, com seus respectivos códigos;
- um mapa da memória dos computadores compatíveis com o Apple II plus com 48 Kbytes de memória;

- um mapa das posições de memória utilizadas pelo Applesoft;
- uma tabela dos códigos ASCII.

Julgamos as informações acima citadas de grande valia para aqueles que pretendem aprofundar seus conhecimentos sobre esse tipo de computador.

I.1. A quem se destina:

Este livro se destina a todos aqueles que já possuam conhecimento elementar sobre a linguagem BASIC. Partindo desse pressuposto, o leitor poderá manipular as informações aqui descritas de maneira muito satisfatória.

Este livro traz informações úteis para iniciantes e 'experts', pois trata de declarações básicas e também avançadas.

* Observações Importantes:

-O Sistema Operacional de Disco utilizado é o D.O.S. 3.3. Não utilize Sistemas Operacionais "envenenados", tais como o Pronto-Dos ou o Hiper-Dos.

-Recomendamos uma minuciosa leitura do apêndice Cuidados com o Computador antes de iniciar a leitura das informações a seguir.

Por que usar PEEK, POKE e CALL:

Estes comandos têm como finalidade acelerar a execução de programas em Applesoft. Algumas dessas declarações não possuem similares na Linguagem Basic do Apple.

I.2. Definições:

As declarações PEEK, POKE e CALL permitem acessar diretamente os endereços de memória. Isso significa que podemos atribuir determinados valores diretamente em posições internas da memória do computador. Assim, obtemos informações processadas de maneira muito mais rápida. Os valores a serem atribuídos às posições de memória podem variar de 0 a 255. Temos, no total, 65.536 posições de memória numeradas.

I.3. POKE:

O comando POKE coloca um valor em uma determinada posição de memória.

Exemplo:

```
POKE 100, 250
```

O valor 250 será armazenado na posição 100 da memória.

* O POKE refere-se apenas à memoria RAM, ou seja, Random Access Memory .

I.4. PEEK:

O comando PEEK recupera um valor de uma determinada posição de memória.

Exemplo:

```
X = PEEK (100)
```


Essa instrução atribuirá à variavel X o valor correspondente à posição de memória 100.

* O PEEK refere-se à memoria RAM ou ROM (ROM=Read Only Memory).

I.5. CALL:

O comando CALL transfere o controle para uma rotina de Linguagem de Máquina.

Exemplo:

```
CALL 62454
```

Esta instrução transfere o controle para a posição 62454 de memória.

* As rotinas de linguagem máquina a serem acessadas pela declaração CALL podem ser intrínsecas da ROM ou serem geradas via software.

- PEEK, POKE E CALL:

Veremos, a seguir, as declarações acima referidas com suas respectivas explicações e eventuais exemplos. Os exemplos foram incluídos de acordo com o grau de dificuldade de interpretação das informações aqui contidas, tendo, portanto, a finalidade de melhor explicar o uso destas instruções.

Algumas instruções terão valor negativo e, portanto, deverão ser precedidas do sinal matemático de subtração (-).

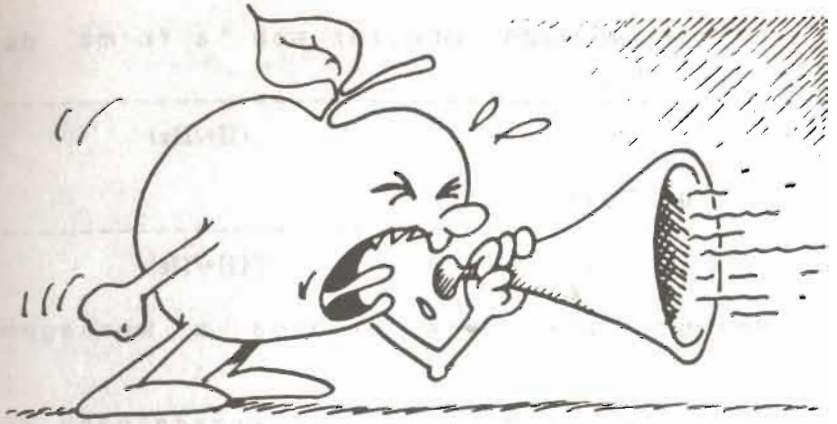
Obs.1: A tecla <RETURN> será solicitada em alguns exemplos. Porém, certos computadores possuem em seu teclado a tecla <RETURN> e outros a tecla <CR>, embora tenham a mesma função.

Obs.2: Todos os exemplos devem ser executados após ser introduzido o Sistema Operacional no computador. Para tal, basta carregar o Disco Mestre que acompanha o seu equipamento. Não se esqueça de digitar a instrução NEW em seguida, objetivando limpar totalmente a memória da máquina.

Obs.3: Muitas instruções contidas neste livro, ao serem utilizadas podem influir em uma próxima instrução a ser testada. Recomendamos, portanto, quando o usuário não conseguir obter o mesmo efeito descrito nos exemplos desligar e religar o computador, com o Disco Mestre do Sistema Operacional de Disco (D.O.S. 3.3).

Obs.4: Ao lado de cada instrução será informado o tipo de computador com o qual a mesma é compatível: a mensagem "(II+/IIe)" indicará que a instrução é compatível com equipamentos da linha APPLE II Plus e APPLE IIe.

II — A Função CALL



1) CALL - 151 (II+/IIe)

Entrar em modo monitor. O modo monitor é a possibilidade que o usuário tem de programar em Linguagem Nativa ou Linguagem Máquina.

- Obs: O modo monitor mostrará um '*' como sinal de pronto. Feito isso, o usuário estará apto a entrar com códigos em Linguagem Máquina. Para retornar ao Applesoft tecle CTRL-C.

---

2) CALL - 155 (II+/IIe)

Entrar em modo monitor fazendo soar um "bip".

O microcomputador Apple possui um alto-falante interno responsável pela emissão de diversos tipos de sons. O "bip" é o som mais tradicional emitido pelo computador, e ao ligá-lo, o usuário obterá esse som.

3) CALL - 167 (II+/IIe)

Entrar em modo monitor sob a forma de texto.

---

4) CALL - 198 (II+/IIe)

Soa um "bip".

---

5) CALL - 211 (II+/IIe)

Soa um "bip" e é exibida a mensagem ERR'.

Essa mensagem é muito utilizada para identificar, no vídeo, que algum tipo de erro ocorreu, chamando a atenção do usuário através de um bip. Ela é muito usada para a indicação de mensagem de erro quando se tem pouco espaço para tal. A mensagem tornou-se quase um padrão internacional para indicar erros. Diversos programas prontos (pacotes de software) a utilizam, tais como: Visicalc, Appleworks e outros.

---

6) CALL - 380 (II+/IIe)

Tem o mesmo efeito que a instrução NORMAL, ou seja, desativa o modo INVERSE ou o modo FLASH.

---

7) CALL - 384 (II+/IIe)

Tem o mesmo efeito da instrução INVERSE.

Essa instrução deve ser utilizada com o intuito de chamar a atenção do usuário. Para desativá-la basta utilizar CALL-380, conforme o ítem 6 acima.

8) CALL - 676 (II+/IIe)

Aguardará que a tecla <RETURN> seja pressionada, após soar um "bip".

Exemplo:

```
10 HOME
20 PRINT "APOS O 'BIP':"
30 PRINT "ESTA MENSAGEM PERMANECERA NA TELA
DO"
40 PRINT "MONITOR ATE QUE <RETURN> SEJA PRE
SSIONADA !"
50 PRINT
60 PRINT "TECLAR <RETURN>..."
70 CALL - 676
80 HOME
90 PRINT "FIM !!!"
```

Essa instrução é muito interessante quando se deseja fazer uma pausa para que algumas instruções importantes, já exibidas no vídeo, possam ser lidas pelo usuário. Chamará a atenção por intermédio de um bip.

---

9) CALL - 678 (II+/IIe)

Aguardará que a tecla <RETURN> seja pressionada (sem soar bip).

Esta instrução tem o mesmo efeito da anterior, sem, porém, emitir o bip.

Exemplo:

```
10 HOME
20 PRINT "ESTA MENSAGEM PERMANECERA NA TELA
DO"
30 PRINT "MONITOR ATE QUE <RETURN> SEJA PRE
SSIONADA !"
40 PRINT
50 PRINT "PRESSIONE <RETURN>..."
60 CALL - 678
70 HOME
80 PRINT "FIM !!!"
```

---

10) CALL - 756 (II+/IIe)

Comando de espera.

Esta instrução tem o mesmo efeito da anterior, porém aceitará que o usuário pressione qualquer tecla para dar sequência ao programa.

Exemplo:

```
10 HOME
20 PRINT "O CALL -756 E UM COMANDO DE ESPER
A."
30 PRINT
40 PRINT"ESTA MENSAGEM PERMANECERA NA TELA"
50 PRINT
60 PRINT "ATE QUE SEJA PRESSIONADA QUALQUER
"
70 PRINT
80 PRINT "TECLA !!!!!!"
90 PRINT
100 CALL -756
110 PRINT "FIM !!!"
```


11) CALL - 856 (II+/IIe)

Esta instrução é conhecida como laço de espera. Sua função é a de retardar a execução de um programa. Pode ser utilizado para uma melhor visualização de resultados quando a demonstração de informações for muito rápida para o acompanhamento humano.

Ou seja, esta tem o objetivo de retardar a execução do programa quando da exibição de uma sequência de mensagens. Desacelerar a execução de um programa ou sequência de dados a ser exibida na tela do monitor pode ser muito útil para a finalidade de demonstração didática de tabelas ou grupos de informações.

Exemplo:

```
10 HOME
20 FOR X=1 TO 160
30 PRINT X
40 NEXT X
50 PRINT
60 PRINT "TEMPO ESTIMADO: 5 SEGUNDOS"
70 PRINT : PRINT "TECLE <RETURN> PARA VER O
MESMO LOOP"
80 PRINT "EM CONJUNTO COM CALL -856:": PRIN
T
90 CALL -756
100 FOR X=1 TO 160
110 CALL -856
120 PRINT X
130 NEXT X
140 PRINT "TEMPO ESTIMADO: 31 SEGUNDOS !"
```

12) CALL - 868 (II+/IIe)

Limpa a linha a partir do cursor.

Esta instrução tem a finalidade de facilitar a edição de dados dentro da tela do monitor. No exemplo, observe que as três solicitações de entrada de dados (NOME, TEL e END) são efetuadas em uma mesma linha, economizando espaço na tela do monitor.

Exemplo:

```
10 HOME
20 HTAB 1:VTAB 1: INPUT"NOME ? ";N$
30 HTAB 1:VTAB 1:CALL - 868: INPUT "TEL.? "
; T$
40 HTAB 1:VTAB 1:CALL - 868: INPUT "END.? "
; E$
50 PRINT
60 PRINT N$
70 PRINT T$
80 PRINT E$
```

---

13) CALL - 912 (II+/IIe)

Eleva todo o texto uma linha (uma linha acima), processo que se denomina SCROLL.

Esta instrução é muito interessante quando o usuário desejar um programa com grande efeito visual. Experimente exibir uma mensagem na tela do monitor e seguí-la de diversos CALL - 912. O efeito será a rolagem do texto dentro do vídeo.

14) CALL - 922 (II+/IIe)

Posiciona o cursor uma linha abaixo da atual.

Esta instrução é muito útil para se tabular um texto dentro da tela do monitor. No exemplo abaixo, obteremos a palavra MONITOR na primeira linha do monitor de vídeo e a palavra AMBAR na terceira linha do vídeo.

Exemplo:

```
10 HOME
20 PRINT
30 PRINT "MONITOR"
40 CALL - 922
50 PRINT "AMBAR"
```

---

15) CALL - 936 (II+/IIe)

Tem o mesmo efeito da instrução HOME, ou seja, limpa o vídeo sem, contudo, alterar o conteúdo da memória ou das variáveis.

---

16) CALL - 958 (II+/IIe)

Limpa a tela a partir do cursor para baixo.

Esta instrução permite a edição de parte da tela do monitor, limpando-a e tendo o cursor *como ponto* de referência. Muito útil quando o programa necessita demonstrar diversas telas com mensagens, este tipo de limpeza parcial da tela do monitor tem grande efeito visual, embelezando o programa.

17) CALL - 998 (II+/IIe)

Posiciona o cursor uma linha acima da atual.

Tem o efeito inverso da instrução CALL-922.

Exemplo:

```
10 HOME
20 PRINT "MICRO"
30 CALL - 998
40 HTAB 6: PRINT "COMPUTADOR"
```

---

18) CALL - 1008 (II+/IIe)

Posiciona o cursor uma coluna à esquerda da atual.

Instrução útil para tabulação dentro do texto apresentado no monitor de vídeo.

Exemplo:

```
10 HOME
20 PRINT "MICROCOMPUTADOR"
30 PRINT "MICRO";: CALL -1008: PRINT "COMPU
TADOR"
```

Observe no exemplo acima, que obteremos duas vezes a palavra MICROCOMPUTADOR, sendo que na segunda linha haverá uma sobreposição de palavras, resultando na palavra "MICRCOMPUTADOR".

19) CALL - 1036 (II+/IIe)

Posiciona o cursor uma coluna à direita da atual.

Esta instrução atua de modo inverso ao da descrita anteriormente no ítem 18, tendo a mesma finalidade de tabulação.

Exemplo:

```
10 HOME
20 PRINT "MICROCOMPUTADOR"
30 PRINT "MICRO";: CALL -1036: PRINT "COMPU
TADOR"
```

Observe no exemplo acima, que obteremos duas vezes a palavra MICROCOMPUTADOR, sendo que na segunda linha haverá um espaço em branco entre as palavras "MICRO" e "COMPUTADOR".

---

20) CALL - 1052 (II+/IIe)

Soa um "bip" tradicional.

Obs: Este comando tem o mesmo efeito do comando número 4.

---

21) CALL - 1184 (II+/IIe)

Exibe, no topo do vídeo, a marca do fabricante (Apple II, por exemplo).

Esta instrução foi inserida neste livro apenas a nível de curiosidade, pois não tem finalidade mais importante.

22) CALL - 1216 (II+IIe)

Tem o mesmo efeito que a instrução GR, ou seja, ativa o modo gráfico de baixa-resolução ou Low-Resolution.

Exemplo:

```
10 HOME
20 CALL -1216
30 COLOR=7
40 HLIN 10,30 AT 10
```

---

23) CALL - 1233 (II+/IIe)

Tem o mesmo efeito que a instrução TEXT, ou seja, desativa o modo gráfico de alta ou baixa resolução ativando o modo texto.

Exemplo:

```
10 HOME
20 GR
30 COLOR=7
40 HLIN 10,30 AT 10
50 VTAB22:PRINT "TECLE <RETURN>...";:GET P$
60 CALL -1233
```

---

24) CALL - 1321 (II+/IIe)

Apresenta os registradores internos:

(A) - Acumulador
(X) - Registrador X
(Y) - Registrador Y
(P) - Program Counter
(S) - Stack Pointer

A apresentação no vídeo será no seguinte formato:

A=98 X=84 Y=D8 P=00 S=B7

Obs: Esta instrução só terá valor para usuários que possuam algum domínio da Linguagem Máquina.

---

25) CALL - 1953 (II+/IIe)

Altera o valor de COLOR no modo GR, ou seja, permite a utilização de diversas cores nos desenhos gráficos.

Todo gráfico, seja em baixa ou alta-resolução, necessita que o usuário defina a cor em que o desenho será exibido na tela do monitor, mesmo que esteja utilizando um monitor do tipo monocromático. Nesse tipo de monitor pode-se observar apenas as variações de tonalidade.

Exemplo:

```
10 HOME
20 GR
30 CALL - 1953
40 HLIN 10,30 AT 10
50 FOR X=1 TO 2000: NEXT X: REM PAUSA PARA
OBSERVACAO !
60 GOTO 30
```

---

26) CALL - 1994 (II+/IIe)

Limpa a tela gráfica de baixa resolução.

Esta instrução é muito útil, pois limpará a tela gráfica de baixa-resolução.

como a instrução HOME utilizada para limpar a tela do monitor no modo texto.

---

27) CALL - 3082 (II+/IIe)

Preenche totalmente a tela de alta resolução com a última cor "HPLOTada". Por ser um preenchimento muito rápido, é utilizado para 'pintar' o fundo da tela do monitor.

Naturalmente, é necessário que o usuário defina a cor a ser utilizada na tela gráfica, bem como a exibição de, no mínimo, um ponto (PLOT) para, em seguida, lançar mão desta instrução.

---

28) CALL - 3086 (II+/IIe)

Limpa a tela gráfica de alta resolução.

Tem o mesmo efeito da instrução CALL - - 1994, descrita acima no item 26. A diferença é que esta limpará a tela gráfica de alta-resolução.

---

29) CALL 42350 (II+/IIe)

Tem o mesmo efeito que o comando CATALOG, ou seja, mostra o diretório do disco, exibindo os nomes dos arquivos, o espaço ocupado por cada arquivo, bem como o tipo de arquivo (Applesoft, Binário, Integer Basic ou arquivo Texto).

O formato de exibição na tela do monitor será o mesmo apresentado pela instrução CATALOG:

A 025 TESTE

onde "A" é o tipo do arquivo (Applesoft), "025" é o número de setores que o programa está ocupando no disco e "TESTE" é o nome do programa.

---

30) CALL 62450 (II+/IIe)

Limpa a tela de alta-resolução, redesenhando toda a tela na cor preta.

Esta instrução é mais lenta do que a CALL - 3086, pois redesenhará a tela do monitor na cor preta.

---

31) CALL 62454 (II+/IIe)

Preenche totalmente a tela de alta resolução com a última cor "HPLOTada", sendo a mesma que CALL - 3082. Esta instrução é muito útil para "pintar" o fundo do monitor de vídeo com o intuito de contrastar com as cores do desenho.

III - A Função PEEK



Para utilizar a função PEEK é necessário precedê-la da instrução PRINT ou de definição variável, ou seja, para utilizar PEEK (-16303), por exemplo, basta digitar:

PRINT PEEK(-16303) e pressionar <RETURN>

---

1) PEEK (-16286) e PEEK (-16287) (II+)

Verifica se os botões dos Paddles "0" e "1" estão sendo pressionados.

O PEEK (-16287) refere-se ao paddle de número 0 e o PEEK (-16286) ao de número 1.

Exemplo:

```
10 HOME
20 X= PEEK (-16287)
30 Y= PEEK (-16286)
40 IF X > 127 THEN GOTO 70
50 IF Y > 127 THEN GOTO 80
60 END
70 PRINT "PADDLE 0 PRESSIONADO !":END
80 PRINT "PADDLE 1 PRESSIONADO !":END
```


No exemplo, os valores dos PEEKs foram armazenados nas variáveis X e Y. Observe que, caso os valores destas variáveis sejam superiores a 127, haverá uma indicação informando a posição do paddle. Caso esteja pressionado, o programa emitirá a mensagem "PADDLE X PRESSIONADO !", onde "X" é o número do paddle.

* Válido apenas para computadores dotados de Paddle.

* Executar o programa acima pressionando o botão de um Paddle de cada vez.

---

2) PEEK (-16303) (II+/IIe)

Altera do modo gráfico, de alta ou baixa-resolução, para o modo texto, ou seja, desativa a tela do monitor que estava em modo gráfico, ativando a tela no modo texto.

---

3) PEEK (-16304) (II+/IIe)

Altera do modo texto para o modo gráfico, ou seja desativa a tela do monitor que estava em modo texto, ativando a tela no modo gráfico. Para voltar ao normal pode-se u-utilizar a instrução acima (PEEK -16303) ou digitar TEXT.

---

4) PEEK (-16336) (II+/IIe)

Emite um sinal sonoro, diferente do "bip" tradicional. Com esta instrução, o usuário poderá diferenciar dois passos importantes dentro de um mesmo programa, utilizando dois "bips" diferentes.

Exemplo:

```
10 HOME
20 PRINT PEEK (-16336)
30 GOTO 20
```

---

5) PEEK (-16384) (II+/IIe)

Pausa sem cursor.

Sempre que o usuário desejar efetuar uma pausa dentro de um programa, para dar tempo de ler uma ou mais mensagens, por exemplo, poderá lançar mão desta instrução. Neste caso, não será exibido o cursor.

Muitas vezes torna-se interessante não exibir o cursor,pois este ocupará um caracter ou uma posição no vídeo, espaço que pode ser útil para a colocação de mais uma letra ou outro caracter qualquer, afim de completar uma mensagem.

Exemplo:

```
10 HOME
20 INPUT"IDADE ? ";I$
30 PRINT
40 PRINT "PRESSIONE QUALQUER TECLA..."
50 PRINT
60 IF PEEK (-16384) < = 26 THEN 60
70 POKE -16368,0
80 GOTO 20
```

Obs: A instrução POKE -16368,0 apresentada na linha 70 é um complemento da linha 60, pois sem ela o programa avançaria parando apenas na "segunda" solicitação de entrada de dados via teclado (um INPUT, por exemplo) pulando uma

primeira solicitação de entrada de dados.
Pode-se substituir o POKE -16368,0 pela instrução GET seguida de uma variável tipo "string" (X$, por exemplo).

---

6) PEEK (44) (II+/IIe)

Apresenta o último ponto "PLOTado". Esta instrução é muito interessante quando utilizada num programa gráfico com o intuito de fazer um desenho tendo suas coordenadas baseadas numa fórmula matemática. Com a instrução acima, pode-se observar a realização do desenho e, ao mesmo tempo, saber qual valor está gerando aquela coordenada na tela do monitor.
Utiliza-se esta em gráfico do tipo baixa-resolução. A modificação da velocidade de apresentação dos caracteres na tela do monitor (SPEED=100) foi modificada (o normal é SPEED=255) para melhor visualização do exemplo. No exemplo abaixo, observe que o desenho gráfico não tem sua velocidade de processamento afetada pela instrução SPEED: só a mensagem "ULTIMO PONTO PLOTADO:" é que será exibida mais lentamente do que o normal.

Exemplo:

```
10 HOME
20 SPEED=100
30 GR
40 COLOR=7
50 FOR X=0 TO 15
```

```
60 PLOT 10,X
70 VTAB 22: PRINT "ULTIMO PONTO PLOTADO: ";
PEEK (44)
80 NEXT X
90 SPEED=255
```

---

7) PEEK (78) (II+/IIe)

Fornece um número aleatório entre "0" e "255", sem seguir nenhuma regra de escolha.

Exemplo:

```
10 Y= PEEK (78)
20 PRINT Y
```

No exemplo acima, o valor aleatório selecionado pela instrução PEEK (78) ficará armazenado na variável "Y", podendo ser manipulado pelo usuário. Números aleatórios são muito apreciados para programas de jogos. Pode-se gerar as coordenadas de um desenho gráfico a partir de números aleatórios.

---

8) PEEK (103) e PEEK (104) (II+/IIe)

Fornece o endereço de início de um programa em Applesoft.

Para conhecedores de Linguagem Máquina esta instrução é muito útil. Tendo o usuário um programa em Applesoft na memória do computador, basta utilizar esta instrução para conhecer o endereço inicial do referido programa.

OBS: Utiliza-se na seguinte fórmula:

PEEK (104) * 256 + PEEK (103)

9) PEEK (105) e PEEK (106) (II+/IIe)

Fornecem o endereço onde se iniciam as variáveis do programa.

Para conhecedores de Linguagem Máquina esta instrução é muito útil. Utilize o programa de conversão Decimal X Hexadecimal descrito no ítem V.8 deste livro.

OBS: Utiliza-se na seguinte fórmula:

PEEK (106) * 256 + PEEK (105)

* Torna-se necessário existir um programa em Applesoft, já processado e com variáveis 'carregadas', na memória do computador para que possa ser fornecido o endereço onde se iniciam as variáveis do programa.

---

10) PEEK (111) e PEEK (112) (II+/IIe)

Indica o endereço inicial das "strings" em forma decimal.

Esta instrução é útil apenas para conhecedores de Linguagem Máquina.

OBS: Utiliza-se na seguinte fórmula:

PEEK (112) * 256 + PEEK (111)

---

11) PEEK (117) e PEEK (118) (II+/IIe)

Apresenta o número da linha em execução no programa.

Esta instrução é muito útil quando se deseja depurar um programa, ou seja, de-

tectar e solucionar erros de lógica. Com esta instrução pode-se acompanhar o desenrolar de um programa, pela tela do monitor, vendo os números das linhas do programa que estão sendo executadas.

Utiliza-se na seguinte fórmula:

```
PEEK (118) * 256 + PEEK (117)
```

Exemplo:

```
10 HOME
20 PRINT "ESTA E A LINHA = "; PEEK (118) *
256 + PEEK (117)
30 END
```

---

12) PEEK (119) e PEEK (120) (II+/IIe)

Apresenta o número da última linha executada no programa.

Utiliza-se esta instrução para acompanhar, dentro de um programa, a última linha que foi executada pelo mesmo, a fim de localizar e solucionar possíveis erros de finalização do programa.

OBS: Utiliza-se na seguinte fórmula:

```
PEEK (120) * 256 + PEEK (119)
```

---

13) PEEK (123) e PEEK (124) (II+/IIe)

Fornece o número da linha DATA que esta sendo lida. Necessita-se de um programa que contenha as instruções READ, DATA e RESTORE.

Esta instrução permite que o usuário acompanhe uma pesquisa dentro de um banco de

dados. Em bancos de dados muito extensos torna-se necessária uma instrução como esta para eventuais alterações de dados, tornando fácil a localização através do número da linha onde os mesmos se encontram.

Exemplo:

```
10 HOME
20 DATA "PEDRO",1,"ANTONIO",2,"PAULA",3
30 DATA "ROBERTO",4,"MARCIA",5
40 DATA "FIM",0
50 INPUT"NOME ? ";N$
60 READ A$,N
70 PRINT PEEK (124) * 256 + PEEK (123)
80 IF A$=N$ THEN GOTO 110
90 IF A$="FIM" THEN PRINT "NAO CONSTA !":
GOTO 40
100 GOTO 60
110 PRINT "NOME=";A$
120 PRINT "NUMERO=";N
130 PRINT
140 GOTO 50
```

---

14) PEEK (129) e PEEK(130) (II+/IIe)

Fornece o nome da última variavel usada no padrão ASCII. Se o nome da variavel tem duas letras, usa-se também PEEK (130).

Exemplo:

```
10 HOME
20 A$="CASA"
30 B$="BOLA"
40 C$="GATO"
50 PRINT "==>> ";PEEK (129)
60 PRINT : PRINT "ULTIMA VARIAVEL UTILIZADA
(EM ASCII): ";PEEK (129)
```

```
70 PRINT: PRINT "ULTIMA VARIAVEL UTILIZADA
EM CARACTER: ";CHR$( PEEK (129))
80 END
```

No exemplo acima, a linha 50 inidicará o número 67, que corresponde a letra "C" no padrão ASCII, já que a última variável utilizada no programa foi C$ (C$="GATO", linha 40).

Na linha 70 temos este código ASCII transformado para o alfabeto normal, originando a letra "C" propriamente dita.

---

15) PEEK (175) e PEEK (176) (II+/IIe)

Fornece o endereço do fim do programa.

Esta instrução torna-se útil para conhecedores de Linguagem Máquina.

OBS: Utiliza-se na seguinte fórmula:

PEEK (176) * 256 + PEEK (175)

Torna-se necessário existir um programa em Applesoft na memória do computador para que possa ser fornecido seu endereço final. Esta instrução é mais utilizada por usuários já familiarizados com a Linguagem Máquina.

---

16) PEEK (202) e PEEK (203) (II+/IIe)

Fornece o endereço do início do programa.

Esta instrução torna-se útil para conhecedores de Linguagem Máquina.

Obs: Utiliza-se na seguinte fórmula:

PEEK (203) * 256 + PEEK (202)

---

17) PEEK (218) (II+/IIe)

Fornece o número da linha onde ocorreu o erro.

Esta instrução é útil na depuração, isto é, na localização de erros dentro de um programa, já que permite a localização da linha onde ocorreu tal erro.

Sugerimos, para melhor aproveitamento desta instrução que ela seja acompanhada da instrução ONERR GOTO, com o intuito de realizar um desvio sempre que ocorrer algum tipo de erro no programa.

Exemplo:

```
10 ONERR GOTO 50
20 HOME
30 GOTO 100: REM LINHA 100 NAO EXISTE !
40 END
50 PRINT "ERRO NA LINHA: ";PEEK (218)
```

---

18) PEEK (222) (II+/IIe)

Fornece o código de um determinado erro ocorrido.

Exemplo:

```
10 ONERR GOTO 50
20 HOME
30 GOTO 100: REM LINHA 100 NAO EXISTE !
40 END
50 PRINT "ERRO ";PEEK (222);"NA LINHA ";
   PEEK (218)
```

OBS: A Tabela dos Códigos de Erro está no Apêndice A. Nesta tabela encontram-se os códigos numéricos e seus respectivos significados.

Caso o usuário queira ser mais específico na indicação do tipo do erro, poderá montar uma tabela no próprio programa, como mostra o exemplo seguinte:

Exemplo:

```
1000 IF PEEK (222) = 0 THEN PRINT "NEXT SEM
FOR"
1010 IF PEEK (222) = 1 THEN PRINT "LINGUAGE
M NAO DISPONIVEL"
   "
   "
   .
   .
   "
   .
   "
   .
   "
   .
   .
1250 IF PEEK (222) = 255 THEN PRINT "CTRL+C
PRESSIONADO"
```

Para que ocorra um desvio para a linha 1000 acima descrita, sempre que algum tipo de erro ocorrer, basta anexar ao programa a seguinte linha:

```
10 ONERR GOTO 1000
```

(ou seja, em caso de erro desvie para a linha 1000).

19) PEEK (224) e PEEK (226) (II+/IIe)

Fornecem as coordenadas do último ponto "HPLOTado".

Exemplo:

```
10 HOME
20 HGR
30 HCOLOR=3
40 HPLOT 1,1 TO 230,1
50 GET K$
60 TEXT: HOME
70 PRINT: PRINT: PRINT
80 PRINT "HPLOT ";PEEK (224);",";PEEK (226)
```

OBS: PEEK (224) : fornece a abscissa
PEEK (226) : fornece a ordenada

* Após executar o exemplo acima, tendo obtido a linha gráfica na tela do monitor, pressione <RETURN> para observar as coordenadas que geraram tal linha.

---

20) PEEK (241) (II+/IIe)

Fornece o valor do SPEED. Este valor regula a velocidade de apresenção dos caracteres na tela do monitor. Esta velocidade pode ser fixada no intervalo de 0 a 255, sendo esta última a velocidade normal.

Usa-se na seguinte fórmula:

```
PRINT 256 - PEEK (241)
```

21) PEEK (978) (II+)

Fornece o tamanho da memória em KBytes.

OBS: Utiliza-se na seguinte fórmula, para computadores com 48 KBytes de memória:

PRINT (PEEK (978) + 35) / 4; "KBYTES"

---

22) PEEK (2040) (II+)

Fornece o valor do último slot acessado, sendo geralmente o de número 6.

Usa-se na seguinte fórmula:

PRINT PEEK (2040) - 192

---

23) PEEK (43624) (II+/IIe)

Fornece o número do último Disk-Drive acessado. Normalmente fornecera 1 ou 2, desde que a interface esteja conectada no slot número 6.

---

24) PEEK (43626) (II+)

Fornece um número (entre 0 e 7) correspondente ao último slot acessado.

---

25) PEEK (43697) (II+/IIe)

Dá o valor do MAXFILES, ou seja, quantos arquivos podem se usar ao mesmo tempo.

---

26) PEEK (44033) (II+/IIe)

Fornece a trilha do diretório, normalmente o número 17 (em hexadecimal).

27) PEEK (46017) (II+/IIe)

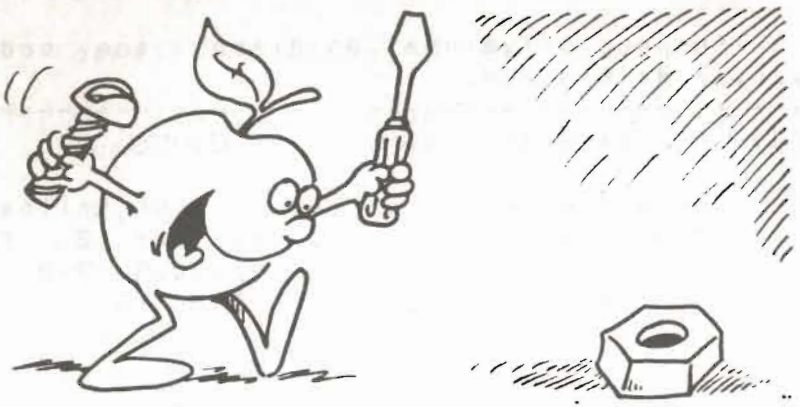
Fornece o volume do disco, que pode variar de 0 a 254.

---

28) PEEK (46064) (II+/IIe)

Fornece o número de setores por trilha. Em D.O.S. 3.3 esse número deve ser 16. Em versões mais antigas como o D.O.S. 3.2, o número de setores por trilha é de 13.

IV — O Comando POKE



O Comando POKE pode ser utilizado no modo direto, simplesmente seguido de <RETURN>, não necessitando, portanto ser precedido da instrução PRINT, como é o caso da função PEEK. Aconselhamos, para perfeito funcionamento das instruções aqui contidas, desligar e religar o computador após cada exemplo testado, a fim de não 'misturar' instruções, já que uma pode influenciar no resultado de outra.

---

1) POKE 32,10 (II+/IIe)

Cria uma janela à esquerda do vídeo. Para anulá-la, utilize a instrução POKE 32,0.

Esta instrução é muito útil quando o usuário necessita demonstrar dados em forma de tabelas. A criação de uma "janela" significa que haverá uma divisão vetical no monitor, onde se pode fixar determinados dados, sem que os mesmos saiam do campo visual do usuário.

Com este POKE cria-se uma mensagem fixa à esquerda do monitor de vídeo. Pode-se anular essa instrução com POKE 32,0 ou pressionando-se a tecla RESET (CTRL - RESET em alguns computadores).

O número 10, à direita da instrução POKE, pode variar entre 0 e 39, conforme o número de colunas que o usuário deseje fixar.

Exemplo:

```
10 HOME
20 POKE 32,10
30 REM
40 REM **********************
50 REM *                     *
60 REM * JANELA ESQUERDA     *
70 REM *                     *
80 REM **********************
90 REM
100 LIST 20,90
110 INPUT "TECLE <RETURN> ";R$
120 POKE 32,0
```

Obs: O monitor de vídeo possui 40 colunas por 24 linhas.

---

2) POKE 33,16 (II+/IIe)

Cria uma janela à direita do vídeo. Para anulá-la, utilize a instrução POKE 33,0.

Esta instrução tem as mesmas características da anterior, com a diferença de criar a janela à direita da tela de vídeo. Da mesma forma que na anterior, pode-se anular a instrução. Para tal, utilize POKE 33,0 ou pressione a tecla RESET (CTRL-RESET

em alguns computadores).

O número 16 à direita da instrução POKE pode variar entre 0 e 39, conforme o número de colunas que o usuário deseje fixar.

Exemplo:

```
10 HOME
20 SPEED=200
30 POKE 33,16
40 PRINT "MICROCOMPUTADOR    ";
50 GOTO 40
```

---

3) POKE 34,10 (II+/IIe)

Cria uma janela horizontal no topo do vídeo. Para anulá-la utilize a instrução POKE 34,0.

Muito útil para montar cabeçalhos fixos. Assim, uma listagem "rolará" no vídeo sem apagar o cabeçalho. Literalmente, o número de linhas do vídeo diminuirá, já que o cabeçalho não fará parte do vídeo normal.

Nesse caso haverá uma divisão horizontal na tela do monitor, onde as linhas determinadas pelo número à direita da instrução serão inapagáveis, mesmo quando se utiliza uma rolagem do vídeo (scroll).

Pode-se anular esta instrução com POKE 34,0 ou pressionando-se RESET (CTRL-RESET em alguns computadores). O número à direita da instrução (4 no exemplo abaixo) pode variar conforme o número de linhas que se deseje fixar para o cabeçalho.

Exemplo 1:

```
10 HOME
20 POKE 34,4
30 PRINT CHR$(4);"CATALOG"
```

No exemplo acima, observe que o cabeçalho do catálogo (DISK VOLUME 254) permanecerá na tela do monitor mesmo que a lista de arquivos contenha diversos nomes (mais de 20, por exemplo).

Exemplo 2:

```
10 HOME
20 POKE 34,4
30 HTAB 12: PRINT "ALFA & CIA LTDA"
40 PRINT
50 HTAB 4: PRINT "R.JORGE PESSOA 345 - PARAISO - SP"
60 PRINT "----------------------------------------"
70 FOR Z=100 TO 200
80 PRINT Z, Z * 2, Z+Z
90 NEXT Z
```

No exemplo acima, observe que o cabeçalho contendo o nome da empresa, o endereço e uma linha tracejada permanecerá na tela do monitor quando a listagem da tabela de números gerada pelo LOOP das linhas 70 à 90 estiver ativa.

---

4) POKE 35,10 (II+/IIe)

Cria uma janela horizontal na parte inferior da tela do vídeo.

Esta instrução tem as mesmas características da anterior, com a diferença de criar a janela no pé da tela do monitor. Da mesma forma que na anterior, pode-se anular esta instrução. Para tal, utilize a instrução POKE 35,0 ou pressione CTRL-RESET (apenas RESET em alguns computadores).

O número à direita da instrução POKE deve variar conforme o número de linhas que o usuário deseje fixar.

Exemplo:

```
10 HOME
20 INVERSE: VTAB 19
30 PRINT SPC (200)
40 VTAB 21: HTAB 16: PRINT "APPLE II"
50 POKE 35,18
60 NORMAL:HOME
```

---

5) POKE 36,X (II+/IIe)

Tabula determinada informação na coluna "X", para X variando de 1 a 40.

A tabulação de dados na tela do monitor é muito útil para a melhor visualização dos dados pelo usuário. O valor de "X" pode variar conforme a necessidade do usuário, inclusive para aqueles que utilizam a interface de 80 colunas para vídeo. Aqueles que utilizam tal interface devem substituir a instrução HOME (linha 10 do exemplo abaixo) por:

```
PRINT CHR$(12)
```


Exemplo:

```
10 HOME
20 POKE 36,10
30 PRINT "COMPUTADOR"
```

---

6) POKE 37,X (II+/IIe)

Tabula determinada informação na linha "X", para X variando de 1 a 23.

Esta instrução tem as mesmas características da anterior, sendo que esta executa a tabulação das informações a partir da linha determinada pelo valor à direita da instrução, onde "X" deve variar de 1 a 23. Apesar desta instrução aceitar uma variação para "X" maior do que 23, não a aconselhamos, pois a apresentação de dados na tela do monitor será prejudicada.

Exemplo:

```
10 HOME
20 POKE 37,10
30 PRINT CHR$(4);"CATALOG"
```

Observe no exemplo acima, que o início do catálogo será na linha 10, conforme foi determinado pelo número à direita da instrução.

---

7) POKE 48,X (II+/IIe)

Em GR, para "X" variando de 0 a 15, define-se a cor de desenhos "PLOTados" na horizontal.

Infelizmente, esta instrução só apresentará o efeito desejado para desenhos plotados na horizontal. A variação da velocidade de apresentação dos caracteres na tela do monitor (20 SPEED=30) tem a finalidade de proporcionar a melhor visualização do efeito deste comando.

Exemplo:

```
10 HOME
20 SPEED=30
30 GR
40 FOR X= 0 TO 15
50 POKE 48,X
60 PLOT X,10
70 VTAB 22: PRINT "COR=";X
80 NEXT X
90 SPEED=255
```

---

8) POKE 50,140 (II+/IIe)

Torna a execução, a listagem e o catálogo invisíveis.

Esta instrução é interessante quando se deseja proteger informações da listagem do programa, a execução do mesmo ou a catalogação de um disco. Como a tecla RESET (ou CTRL-RESET em alguns computadores) pode anular esta instrução, para uma proteção de dados mais eficiente, sugerimos combiná-la com as instruções que anulam a tecla RESET. Veja o ítem 16 deste capítulo para tais instruções.

Exemplo:

```
10 HOME
20 PRINT "TORNA TANTO A EXECUCAO QUANTO"
30 PRINT "A LISTAGEM INVISIVEIS !"
40 POKE 50,140
50 PRINT "COMPUTADOR"
```

---

9) POKE 50,X (II+/IIe)

Define a forma com a qual as informações serão exibidas na tela do monitor.

Para X=63 => INVERSE

- a próxima informação a ser exibida na na tela do monitor será apresentada no modo invertido, ou seja, em fundo branco com as letras em preto.

Para X=127 => FLASH

- a próxima informação a ser exibida na tela do monitor será apresentada no modo piscando, ou seja, em fundo branco com as letras em preto e vice-versa o que nos dará a noção de 'piscando'.

Para X=255 => NORMAL (letra normal)

- a próxima informação a ser exibida na tela do monitor será apresentada no modo normal, ou seja, em fundo preto e as letras em branco.

Exemplo:

```
10 HOME
20 POKE 50,63
30 PRINT "CORINTHIANS"
40 PRINT
50 POKE 50,127
60 PRINT "SAO PAULO"
70 PRINT
80 POKE 50,255
90 PRINT "PORTUGUESA"
```

---

10) POKE 214,X (II+/IIe)

Tranforma qualquer comando em RUN, desde que "X" seja maior do que "127"; não permitindo posterior listagem do programa.

Com esta instrução ativa, a execução do programa poderá ser interrompida a qualquer momento, bastando, para tal, pressionar a tecla RESET (ou CTRL-RESET). Neste caso, as duas únicas instruções que poderão ser executadas serão CATALOG e PR#(II PLUS/IIE)6 (REBOOT). Qualquer outra instrução transformar-se-á em RUN, ou seja, processará o programa novamente. Esta instrução é ótima para proteger programas contra listagens.

Para voltar ao normal, utilize POKE 214,0.

---

11) POKE 216,X (II+/IIe)

Anula a instrução ONERR GOTO, desde que "X" seja menor do que "128", ou seja, desativa o desvio que a instrução ONERR GOTO executará caso ocorra algum tipo de erro dentro do programa.

12) POKE 241,256-X (II+/IIe)

Modifica o valor de SPEED, instrução que determina a velocidade de apresentação dos caracteres na tela do monitor.

Exemplo:

Para SPEED=255

POKE 241,1 (X=255)

Obs: A instrução SPEED pode variar de valor no intervalo de 1 a 255.

---

13) POKE 242,128 (II+/IIe)

Monitorador de programa.

Esta instrução tem o mesmo efeito da instrução TRACE, originando uma monitoração do programa e mostrando os números das linhas que estão sendo executadas no mesmo. Para anular esta instrução, basta digitar NOTRACE. Observe que a execução do programa será feita de modo normal; a única diferença será a demonstração, na tela do monitor, do símbolo "#" seguido do número da linha que o programa está executando.

---

14) POKE 243,32 (II+/IIe)

Retorna o STRING de maiúsculo para minúsculo.

Alguns computadores não podem gerar letras minúsculas originariamente. Nesse caso as letras minúsculas serão substituídas por diversos caracteres, tais como "*", "(", ")", etc, sendo impossível "decifrar" a mensagem exibida.

Para anular este comando, basta digitar POKE 243,0.

Exemplo:

```
10 A$="PINDAMONHANGABA"
20 POKE 243,32
30 PRINT A$
```

---

15) POKE 245,0: POKE 242,128 (II+/IIe)

Tem a mesma função do comando TRACE, que é usado para rastreameto de programas, informando os números das linhas pelas quais o programa está passando ao ser executado.

Esta é uma variação da instrução apresentada no ítem 13 deste capítulo.

Para anular este comando, basta digitar NOTRACE.

---

16) POKE 1009,76: POKE 1010,241: (II+/IIe)
POKE 1011,165: POKE 1012,166

Proteção contra RESET (para qualquer capacidade de memória, ou seja, o computador pode estar conectado a expansões de memória). Anexar estas instruções no início do programa.

Esta instrução é útil para a proteção de programas contra listagem, pois não permite parar a execução do mesmo afim de listá-lo.

Obs: Ao ser pressionada a tecla <RESET> haverá um REBOOT, ou seja, o efeito será o mesmo de se ter desligado e, em seguida, religado o computador.

17) POKE 1010,102: POKE 1011,213: (II+/IIe)
POKE 1012,112

Transforma RESET em RUN. Anexar estas instruções no início do programa.

Com esta instrução, ao ser pressionada a tecla RESET (ou CTRL-RESET em alguns computadores), o programa se auto-executará a partir de seu início, não efetuando a parada de execução do mesmo.

Para anular utilize: POKE 1010,191: POKE 1011,157: POKE 1012,56.

---

18) POKE 1010,30: POKE 1011,60: (II+/IIe)
POKE 1012,90

Proteção contra RESET.

Instrução idêntica à citada no tópico número 16, executará novamente o programa a partir de seu início.

Esta instrução servirá para aqueles usuários que possuam computadores com qualquer capacidade de memória, pois suportará qualquer interface de expansão da mesma. Sugerimos anexar este conjunto de instruções no início do programa.

Obs: Ao ser pressionada a tecla <RESET> haverá um REBOOT, ou seja, o efeito será o mesmo de se ter desligado e, em seguida, religado o computador.

---

19) POKE 1013,76: POKE 1014,102: (II+/IIe)
POKE 1015,213

Ao ser digitado um '&' seguido de <RETURN> este será interpretado como RUN, ou seja, executará o programa novamente a

partir de seu início. Como a instrução RUN é muito utilizada em programação, tornou-se útil definí-la em apenas uma tacla para economizar tempo. A escolha do símbolo "&" para definir a instrução RUN foi muito feliz, pois trata-se de uma tecla pouco utilizada.

---

20) POKE 1013,76: POKE 1014,105: (II+/IIe)
POKE 1015,255

Ao ser digitado um '&' seguido de <RETURN>, este será interpretado como CALL-151, ou seja, entrará em modo Monitor. Para retornar do modo Monitor ao Applesoft tecle CTRL-C seguido da tecla <RETURN> (ou CR em alguns computadores).

---

21) POKE 1013,76: POKE 1014,165: (II+/IIe)
POKE 1015,214

Ao ser digitado um '&' seguido de <RETURN> este será interpretado como LIST.

---

22) POKE 1013,76: POKE 1014,228: (II+)
POKE 1015,251

Ao ser digitado um '&' seguido de <RETURN>, este será interpretado como "BIP".

---

23) POKE 1013,76: POKE 1014,88: (II+/IIe)
POKE 1015,252

Ao ser digitado um '&' seguido de <RETURN>, este será interpretado como HOME, limpando a tela do monitor.

24) POKE 1024,X até (II+/IIe)
POKE 2039,X

Imprime em qualquer posição da tela do monitor (definida por um número entre 1024 e 2039) o caracter correspondente ao código ASCII definido por X.

Esta instrução é ótima para criptografar seu nome ou o de sua empresa, a fim de escondê-lo dentro do programa. O usuário comum não saberá que esse grupo de instruções, na verdade, se compõe de apenas um nome ou uma mensagem codificada.

Observe que a cada intervalo de 40 números as informações aparecerão no vídeo em linhas diferentes (de 8 em 8 linhas).

Exemplo 1:

```
10 HOME
20 POKE 1293,67
30 POKE 1294,79
40 POKE 1295,77
50 POKE 1296,80
60 POKE 1297,85
70 POKE 1298,84
80 POKE 1299,65
90 POKE 1300,68
100 POKE 1301,79
110 POKE 1302,82
```

Obs: Os números à direita do POKE são códigos ASCII (67=C, 79=O, etc). No exemplo acima, teremos formado a palavra COMPUTADOR.

Exemplo 2:

```
10 HOME
20 FOR E = 1024 TO 2039
30 POKE E,255
40 NEXT E
```

No exemplo acima a tela do monitor vai sendo preenchida com um caractere gráfico (de código 255) desde o endereço 1024 (primeira posição da linha 1) até o 2039 (última posição da última linha). Observe a sequência de preenchimento da tela!

---

25) POKE 2049,1 (II+/IIe)

Faz a listagem demonstrar apenas a primeira linha do programa.

---

26) POKE 40286,35: POKE 40287,216 (II+)

Proteção contra RESET, só para computadores com 48 KBytes. O CTRL-C continua funcionando para brecar o programa.

Exemplo:

```
10 ONERR GOTO 80
20 POKE 40286,35: POKE 40287,216
30 HOME
40 PRINT "INTERROMPIDO ";I;" VEZES !"
50 GOTO 50
60 ONERR GOTO 80
70 POKE 40286,35: POKE 40287,216
80 I=I+1: GOTO 10
```


Obs: Execute e pressione <RESET> para observar o efeito. Este código entende o pressionar do <RESET> como erro ! A cada pressão no <RESET> o número de interrupções demonstrado no vídeo será incrementado em um.

A tabela contendo as Mensagens de Erro do Sistema Operacional de Disco Versão 3.3 encontra-se no Apêndice A.

---

27) POKE 42768,234: POKE 42769,234: (II+/IIe)
POKE 42770,234

Trava o computador quando ocorre algum erro. Além de não ser exibida a mensagem de erro, a liberação do cursor só ocorre se pressionarmos CTRL-RESET.

---

28) POKE 44033,X (II+/IIe)

Possibilita a escolha da trilha onde será armazenado o diretório, para "X" variando de 3 a 35. A trilha normal de armazenamento do diretório é a 17 (ou 11 em hexadecimal). Porém, pode-se armazenar o diretório em qualquer trilha do disco, da 3 à trilha 35. E' possível, também, termos dois diretórios em um mesmo disco. Nesse caso, a ordem em que um ou outro diretório será acessado dependerá de instruções fornecidas pelo usuário em modo direto ou programado.

Muitas vezes um programa pronto não possibilita sua "CATALOGação", ou seja, ao catalogarmos o disco, recebe-se uma mensagem de ERRO E/S (erro de entrada ou saída). Em alguns casos, o programa foi removido da trilha original para uma outra qualquer.

Para "vasculhar" um disco, trilha a trilha, utilize o seguinte programa:

```
10 ONERR GOTO 70
20 FOR X=3 TO 35
30 POKE 44033,X
40 PRINT CHR$(4); "CATALOG"
50 IF X=35 THEN END
60 INPUT "(RETURN) PARA CONTINUAR: ";RT$
70 NEXT X
```

No programa acima temos um laço variando o valor de X entre 3 e 35, que são os números das trilhas a serem analisadas. A instrução POKE 44033 fará a mudança de uma trilha para outra. Na linha 40 temos a instrução que possibilitará a catalogação de cada trilha. Quando for feita uma catalogação em uma trilha que nada contenha, ocorrerá um erro, daí utilizarmos a instrução da linha 10, que executará um desvio para a linha 70, dando continuidade ao processo.

---

29) POKE 44452,24: (II+/IIe)
POKE 44605,23

Apresenta os arquivos do diretório, mostrando 20 nomes de cada vez.

Estas instruções devem ser seguidas de CATALOG para a exibição dos arquivos gravados em disco. Para utilizar a instrução CATALOG dentro de um programa, siga a seguinte sintaxe:

```
PRINT CHR$(4);"CATALOG"
```


30) POKE 44505,234: (II+/IIe)
POKE 44506,234

Apresenta os arquivos já deletados (apagados) do disco.

Também pode-se obter o mesmo efeito utilizando POKE 44506,0. Para normalizar o catálogo, utilize POKE 44506,74.

---

31) POKE 44567,X (II+/IIe)

Possibilita padronizar o catálogo, especificando o número de caracteres mais um (x+1) dos nomes dos arquivos demonstrados, para "X" variando de 0 a 29.

Exemplo:

```
POKE 44567,5
CATALOG
```

Caso "X" seja 0 (zero) o catálogo mostrará apenas a primeira letra do nome de cada arquivo gravado no disco.

---

32) POKE 44578,234: POKE 44579,234: (II+)
POKE 44580,234

Altera a formatação da tela desordenando a apresentação do CATALOG.

Exemplo:

```
10 HOME
20 POKE 44578,234: POKE 44579,234:
   POKE 44580,234
30 PRINT CHR$(4);"CATALOG"
```

33) POKE 44596,234: POKE 44597,234: (II+/IIe)
POKE 44598,234

Cancela os RETURNs no catálogo, listando todos os nomes dos arquivos sem pausa, ou seja, exibe todo o catálogo do disco de uma só vez. Normalmente, quando mais de 18 (dezoito) arquivos são gravados em um mesmo disco e o usuário solicita uma lista do mesmo, são exibidos 18 nomes de arquivos cada vez. Para passar dos primeiros 18 arquivos iniciais para os seguintes torna-se necessário que o usuário tecle <RETURN> (ou CR em alguns computadores).

Exemplo:

```
10 HOME
20 POKE 44596,234: POKE 44597,234:
   POKE 44598,234
30 PRINT CHR$(4);"CATALOG"
```

---

34) POKE 44599,234: POKE 44600,234 (II+)

Provoca pausa em cada arquivo do CATALOG, ou seja, tornar-se-á necessário, dentro do catálogo do disco, pressionar a tecla <RETURN> para a exibição de cada nome do arquivo seguinte.

Exemplo:

```
10 HOME
20 POKE 44599,234: POKE 44600,234:
30 PRINT CHR$(4);"CATALOG"
```

Tecle <RETURN> para que seja mostrado o nome seguinte do arquivo.

35) POKE 44611,X (II+/IIe)

Possibilita padronizar o número de setores e o volume, especificando o número de caracteres mais um (X= No.de caract.+1) com X variando de 2 a 26.

Caso o usuário queira reduzir ou aumentar o campo onde se encontram as informações relativas ao número de setores que os arquivos estão ocupando no disco, bem como o número do volume do disco, deve lançar mão desta instrução. Observe que se um determinado arquivo estiver ocupando, por exemplo, 11 setores de um disco e o usuário padronizar a exibição dos setores para 1 (um) caracter (POKE 44611,0), esse arquivo apresentará um caracter especial no lugar do número de setores, pois com apenas um dígito não poderá exibir corretamente o número de setores (11) que o mesmo está ocupando no disco.

Esses caracteres especiais podem variar ("?" ou ";" ou ":" , etc) sem regras lógicas, pois apenas indicam que os números dos setores apresentados não são reais.

Exemplo:

```
POKE 44611,10
CATALOG
```

---

36) POKE 45991,X (II+/IIe)

Possibilita alteração da indicação de Arquivo Texto (T) para qualquer outra letra ou símbolo.

Exemplo:

```
POKE 45991,48
CATALOG
```

Obs: Todos os arquivos em modo Texto terão prefixo "0", já que o código escolhido foi "48".

---

37) POKE 45992,X (II+/IIe)

Possibilita alteração da indicação de Arquivo Integer Basic (I) para qualquer outro símbolo.

Exemplo:

```
POKE 45992,67
CATALOG
```

Obs: Todos os arquivos em Linguagem Integer Basic terão prefixo "C", já que o código escolhido foi "67".

---

38) POKE 45993,X (II+/IIe)

Possibilita alteração da indicação de Arquivo Applesoft (A) para qualquer outro símbolo.

Exemplo:

```
POKE 45993,69
CATALOG
```

Obs: Todos os arquivos em Linguagem BASIC terão prefixo "E", já que o código escolhido foi "69".

39) POKE 45994,X (II+/IIe)

Possibilita alteração da indicação de Arquivo Binário (B) para qualquer outro símbolo.

Exemplo:

```
POKE 45994,72
CATALOG
```

Obs: Todos os arquivos em Binário terão prefixo "H", já que o código escolhido foi "72".

---

40) POKE X,Y (II+/IIe)

Possibilita alteração do cabeçalho do CATALOG, sendo que:

=> "X" = deve variar de 45999 a 46010.

=> "Y" = pode variar de 0 a 255 (ASCII).

Ao catalogar um disco, normalmente obtém-se a seguinte mensagem:

DISK VOLUME 254

Pode-se alterar esta informação para qualquer outra, utilizando estas instruções.

O "X" indicará a posição em que cada caracter (Y) ficará situado no cabeçalho.

A ordem de exibição desses caracteres que utilizam o código ASCII será inversa. Cada caracter alterado situar-se-á a partir da última letra ou número do cabeçalho.

Esta instrução é muito utilizada para a personalização do cabeçalho do catálogo.

Exemplo:

```
10 HOME
11 POKE 46010, ASC ("E")
12 POKE 46009, ASC ("D")
13 POKE 46008, ASC ("I")
14 POKE 46007, ASC ("T")
15 POKE 46006, ASC ("O")
16 POKE 46005, ASC ("R")
17 POKE 46004, ASC ("A")
18 POKE 46003, ASC (" ")
19 POKE 46002, ASC ("A")
20 POKE 46001, ASC ("L")
21 POKE 46000, ASC ("E")
22 POKE 45999, ASC ("P")
23 POKE 45998, ASC ("H")
```

---

41) POKE 44452,255:
POKE 44513,67 (II+/IIe)

Cataloga apenas os arquivos travados, ou seja, aqueles em que foi utilizada a instrução LOCK.

Exemplo:

```
10  HOME
20  POKE 44452,255: POKE 44513,67
30  PRINT  CHR$ (4);"CATALOG"
40  END
```

Para retornar ao catálogo normal, basta executar o programa a seguir:


```
10  HOME
20  POKE 44452,22: POKE 44513,2
30  PRINT  CHR$ (4);"CATALOG"
40  END
```

---

42) POKE 49385,0 (II+/IIe)

Aciona o Leitor de Disco (Disk Drive) sem ler ou gravar qualquer informação em disco.

Para a utilização de um Head-Cleaner, isto é, um disco especial para a limpeza de cabeçotes de disk drive é necessário acionar o drive para que o disco de limpeza possa girar e, assim, limpar o cabeçote. A instrução acima é ideal para essa tarefa, já que o leitor de disco permanecerá acionado sem que ocorra gravação ou leitura de dados e não provocará uma mensagem de erro.

Esta instrução acionará o leitor de disco número um. Para executar a mesma tarefa com o leitor de disco número dois, basta catalogar um disco inicializado no mesmo e digitar a mesma instrução

POKE 49385,0.

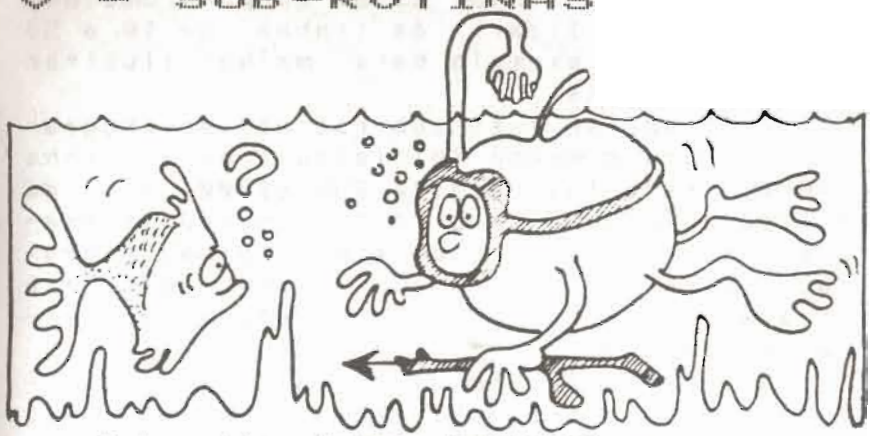
Para desativar esta instrução tecle RESET (ou CTRL-RESET em alguns computadores).

Obs: Anula com POKE 49384,0. Pode-se continuar trabalhando com o computador, mesmo com o leitor de disco acionado.

43) POKE -16302,0 (II+)

Após uma instrução HGR, retira as 4 linhas de texto, deixando toda tela disponível para os gráficos.

V — SUB-ROTINAS



Sub-rotina é uma parte de um programa que tem finalidade específica. Damos, a seguir uma pequena relação de sub-rotinas que anexadas ou não a outros programas, poderão auxiliar o programador.

---

1) Colocação da marca personalizada em programa:

```
10 HOME
20 PRINT : PRINT "PARA EXECUTAR:": PRINT:
PRINT
30 PRINT TAB (7); "- RUN 60000 (INICIALMENT
E)": PRINT
40 PRINT TAB (7); "- EM SEGUIDA LISTAR !!!"
50 END
60000 Y=PEEK (105) + PEEK (106) * 256
60010 FOR X= Y - 256 TO Y
60020 IF PEEK (X)+ PEEK (X+1)* 256 <> 63999
THEN NEXT
60030 POKE X,255: POKE X+1,255
60040 DEL 60000,60040: END
63999 REM DIGITE SEU NOME NESTE REM !!!!
```

O usuário deve introduzir a rotina da linha 60000 à linha 63999 no programa que deseja personalizar. As linhas de 10 a 50 são apenas um exemplo para melhor ilustrar essa sub-rotina.

Após anexar essa sub-rotina ao programa, execute o mesmo iniciando-o pela linha 60000. Para tal, digite RUN 60000, seguido da tecla RETURN. Isso feito, aguarde a execução do programa. Ao término dessa execução, liste o programa. Observe que será criada uma linha 65535 indeletável, ou seja, inapagável.

Na instrução REM da linha 63999, substitua 'DIGITE SEU NOME NESTE REM !!!!' pelo seu nome ou marca a ser definitivamente registrada.

---

2) Gerando sons:

```
10 HOME
20 PRINT "ENTRE VALORES DE 0 A 255": PRINT
30 DATA 160,0,1,174,0,3,173,48,192
40 DATA 136,208,5,206,1,3,240,5
50 DATA 202,208,245,240,237,96,256
60 Y = 777
70 READ X
80 IF X = 256 THEN GOTO 120
90 POKE Y,X
100 Y = Y + 1
110 GOTO 70
120 INPUT "DE .. ";U
130 INPUT "ATE'. ";D
140 FOR P = U TO D
150 PRINT "TOM = ";P
160 POKE 768,P: POKE 769,255
170 CALL 770: NEXT P
180 GOTO 120
```


Obs: POKE 769,X = Possibilita que a rotina de som torne-se mais lenta. Diminuindo o valor de X, ela será mais rápida (variar X de 0 a 255).

---

3) Tabela de Instruções do Applesoft:

Apresenta uma tabela contendo as instruções do Applesoft.

```
10 HOME
20 FOR A = 53456 TO 53860
30 PRINT "CODIGO: ";A;" = "; PEEK (A);
TAB (30); CHR$(PEEK(A))
40 PRINT "-----------------------------------"
50 NEXT A
```

As instruções serão demonstradas verticalmente, letra a letra:

```
CODIGO: 53456 = 69                    E
-----------------------------------------
CODIGO: 53457 = 78                    N
-----------------------------------------
CODIGO: 53458 = 68                    D
-----------------------------------------
CODIGO: 53459 = 70                    F
-----------------------------------------
CODIGO: 53460 = 79                    O
-----------------------------------------
CODIGO: 53461 = 82                    R
-----------------------------------------
```

```
CODIGO: 53856 = 78                N
CODIGO: 53857 = 69                E
CODIGO: 53858 = 88                X
CODIGO: 53859 = 84                T
CODIGO: 53860 = 32
```

Observe na vertical à direita, que as intruções (END, FOR...NEXT) são demonstradas na vertical. Os números exibidos após o sinal de igual são códigos ASCII e referem-se às letras que compõem as instruções do Applesoft.

---

4) Tabela de Comandos D.O.S. 3.3:

Apresenta uma tabela contendo os comandos do Disk Operating System, Versão 3.3.

```
10 HOME
20 FOR A = 43140 TO 43271
30 PRINT "CODIGO: ";A;" = "; PEEK (A);
TAB (30); CHR$(PEEK(A))
40 PRINT "-----------------------------------"
50 NEXT A
```

As instruções serão demonstradas verticalmente, letra a letra:


```
CODIGO: 43140 = 73                I
CODIGO: 43141 = 78                N
CODIGO: 43142 = 73                I
CODIGO: 43143 = 212               T
CODIGO: 43144 = 76                L
CODIGO: 43145 = 79                O
CODIGO: 43146 = 65                A
CODIGO: 43271 = 217               D
    .       .       .             .
    .       .       .             .
    .       .       .             .
    .       .       .             .
    .       .       .             .
CODIGO: 43266 = 86                V
CODIGO: 43267 = 69                E
CODIGO: 43268 = 82                R
CODIGO: 43269 = 73                I
CODIGO: 43270 = 70                F
CODIGO: 43271 = 217               Y
```

Observe na vertical à direita, que as intruções (INIT, LOAD...VERIFY) são demonstradas na vertical. Os números exibidos após o sinal de igual são códigos ASCII e referem-se às letras que compõem as instruções do Applesoft.

---

5) Tabela das Mensagens de Erro:

Apresenta uma tabela contendo as Mensagens de Erro.

```
10 HOME
20 FOR A = 43378 TO 43582
30 PRINT "CODIGO: ";A;" = "; PEEK (A);
TAB (30); CHR$(PEEK(A))
40 PRINT "----------------------------------------"
50 NEXT A
```

As instruções serão demonstradas verticalmente, letra a letra:

```
   .     .     .          .
   .     .     .          .
   .     .     .          .
   .     .     .          .

----------------------------------------
CODIGO: 43468 = 69              E
----------------------------------------
CODIGO: 43469 = 82              R
----------------------------------------
CODIGO: 43470 = 82              R
----------------------------------------
CODIGO: 43471 = 79              O
----------------------------------------
CODIGO: 43472 = 32
----------------------------------------
CODIGO: 43473 = 69              E
```



```
----------------------------------------
CODIGO: 43474 = 47              /
----------------------------------------
CODIGO: 43475 = 83              S
   .     .     .          .
   .     .     .          .
   .     .     .          .
```

Observe na vertical à direita, que a instrução (ERRO E/S) é demonstrada na vertical. Os números exibidos após o sinal de igual são códigos ASCII e referem-se às letras que compõem as instruções do Applesoft.

---

6) Tabela do Catálogo:

```
10 HOME
20 FOR A = 45991 TO 49230
30 PRINT "CODIGO: ";A;" = "; PEEK (A);
TAB (30); CHR$(PEEK(A))
40 PRINT "----------------------------------------"
50 NEXT A
```

Obs: Apresenta uma tabela contendo os nomes dos arquivos gravados em disco, desde que estejam na memória do computador. Arquivos gravados antes do computador ser desligado não estarão mais registrados e, portanto, não aparecerão com esta instrução.
Para testar esta instrução, grave um programa qualquer em disco e execute a sub-rotina acima.

7) Rotina HIDDEN:

Caracteres HIDDEN são aqueles utilizados como de controle, por exemplo o CTRL-D, CTRL-S, CTRL-A, CTRL-G, etc. Esses caracteres podem fazer parte tanto de um programa, quanto do diretório (catálogo do disco). Ao gravar um programa em disco, necessitamos de um nome para o mesmo. Introduzindo-se entre as letras deste nome alguns caracteres HIDDEN, estes não aparecerão no catálogo do diretório. Portanto, apenas quem conhece a posição destes caracteres de controle poderá carregar/executar tais programas.

Esta rotina permite catalogar arquivos gravados em disco, cujos nomes contenham caracteres de controle "escondidos". Da mesma forma, se esses caracteres especiais fizessem parte do corpo de um determinado programa, poderão tornar-se "visíveis" com um simples LIST, através da rotina abaixo.

```
10  HOME
20  DATA  201,141,240,21,201,136
30  DATA  240,17,201,128,144,13
35  DATA  201,160,176,9,72,132
40  DATA  53,56,233,64,76,249
50  DATA  253,76,240,253
60  FOR I = 768 TO 768 + 27
70  READ V: POKE I,V: NEXT I
80  POKE 54,0: POKE 55,3
90  CALL 1002
```

Experimente gravar um programa em disco com o seguinte nome:

SAVE TESTêANDO

onde o "ê" significa CTRL-D. Dê o CATALOG no mesmo disco e o nome do arquivo aparecerá comò "TESTANDO", portanto, não estaremos vendo o caracter de controle CTRL-D. Tente agora carregar o mesmo programa utilizando a instrução LOAD TESTANDO. Observe que o computador emitirá a mensagem "ARQUIVO NAO ENCONTRADO" ou "FILE NOT FOUND". Tente a mesma operação, porém incluindo o CTRL-D entre as letras "T" e "A", conforme foi gravado. Observe que agora o carregamento do programa deu-se de forma normal. Para encontrar esses caracteres de controle Hidden, basta utilizar a sub-rotina acima.

---

8) Relação Caracter X Código ASCII

Nesta rotina, entrando com qualquer caracter, obtemos o correspondente código ASCII.

```
10  HOME
20  IF  PEEK ( - 16384) < 128 THEN 20
30  A =  PEEK ( - 16384)
40  PRINT  CHR$ (A);".....";A
50  POKE  - 16368,0
60  GOTO 20
```

---

9) Programa de Conversão:

HEXADECIMAL < == > DECIMAL

Este programa executa a conversão de números decimais para hexadecimais e vice-versa.

```
3   ONERR  GOTO 70
4   GOTO 3000
5   HOME
10  INPUT "DECIMAL = ";DC$
11  DC = VAL (DC$)
15  IF DC$ = "S" THEN 95
16  IF DC$ = "M" THEN 3000
20  AX =  INT (DC / 256)
30  POKE 61,AX
40  POKE 60,DC - 256 * AX
50  CALL  - 622: PRINT " HEXADECIMAL"
55  PRINT "-----------------------------------
-------"
60  PRINT : GOTO 10
70  CALL  - 1052: PRINT : VTAB 22: PRINT
"DIGITE NOVAMENTE !!!"
80  GOTO 3000
90  GOTO 5
95  HOME : VTAB 5: HTAB 10: PRINT "SAIR ?
(S/N)"
100  POKE  - 16368,0: WAIT  - 16384,128:A$
= CHR$ ( PEEK ( - 16384) - 128): POKE  -
16368,0
110  IF A$ = "N" THEN  RUN
115  IF A$ = "S" THEN : HOME : END
120  IF A$ <  > "S" OR A$ <  > "N" THEN
CALL  - 198: GOTO 100
130  END
140  HOME
150  INPUT "HEXADECIMAL = ";Z$
160  IF Z$ = "S" THEN GOTO 95
170  IF Z$ = "M" THEN GOTO 3000
180  A =  LEN (Z$): IF A = 0 OR A > 4 OR Z$
= "" THEN  CALL  - 198: CALL  - 198:
GOTO 140
190   IF  LEN (Z$) = 1 THEN Z$ = "000" +
(Z$)
200   IF  LEN (Z$) = 2 THEN Z$ = "00" +
(Z$)
```



```
210   IF  LEN (Z$) = 3 THEN Z$ = "0" + (Z$)
220  Z1$ =  MID$ (Z$,1,1)
230  Z2$ =  MID$ (Z$,2,1)
240  Z3$ =  MID$ (Z$,3,1)
250  Z4$ =  MID$ (Z$,4,1)
260   IF Z1$ >  = "A" AND Z1$ <  = "F" THEN
Z1 =  ASC (Z1$) - 55
270   IF Z2$ >  = "A" AND Z2$ <  = "F" THEN
Z2 =  ASC (Z2$) - 55
280   IF Z3$ >  = "A" AND Z3$ <  = "F" THEN
Z3 =  ASC (Z3$) - 55
290   IF Z4$ >  = "A" AND Z4$ <  = "F" THEN
Z4 =  ASC (Z4$) - 55
300   IF Z1$ >  = "0" AND Z1$ <  = "9" THEN
Z1 =  ASC (Z1$) - 48
310   IF Z2$ >  = "0" AND Z2$ <  = "9" THEN
Z2 =  ASC (Z2$) - 48
320   IF Z3$ >  = "0" AND Z3$ <  = "9" THEN
Z3 = ASC (Z3$) - 48
330   IF Z4$ >  = "0" AND Z4$ <  = "9" THEN
Z4 = ASC (Z4$) - 48
340 HX = Z1 * 4096 + Z2 * 256 + Z3 * 16 +
Z4
350   PRINT : PRINT HX; "- DECIMAL"
360   PRINT "-----------------------------------
----------" : GOTO 150
3000 REM  ---  MENU  ---
3001 CLEAR
3010 HOME
3020 VTAB 4: HTAB 13: PRINT "CONVERSOR"
3030 VTAB 7: PRINT  TAB( 5); "1. DECIMAL P/
HEXADECIMAL"
3040 VTAB 9: PRINT  TAB( 5);
"2. HEXADECIMAL P/ DECIMAL"
3050 PRINT : PRINT : PRINT TAB( 5);"ESCOLHA
.. :";: GET K$
3060 IF  VAL (K$) = 1 THEN GOTO 5
3070 IF  VAL (K$) = 2 THEN GOTO 140
3080 CALL  - 198: GOTO 3010
```

Ao executar este programa, após a escolha no menu, tecle:

<S> para sair do programa e retornar ao Applesoft.

<M> para retornar ao menu principal.

Este programa será muito útil para aqueles usuários que desejarem aprofundar-se em Linguagem Máquina. Essa linguagem utilizará seus códigos em um tipo de numeração denominado Hexadecimal. Para encontrar o código equivalente ao hexadecimal em números decimais, poderá lançar mão deste programa para executar tal conversão.

No Apêndice "F" temos as instruções do Applesoft e seus respectivos valores em decimal. Ao listarmos um programa em Linguagem Máquina, poderemos decodificá-lo, analisando os seus passos, utilizando a tabela deste apêndice (F) em conjunto com a subrotina de conversão mostrada acima. Para maiores detalhes, observe o exemplo apresentado no Apêndice "F"

---

10) Rotina Applesoft X Arquivo Texto

Esta rotina permitirá transformar seu programa de linguagem Applesoft para um arquivo do tipo Texto. A transformação permitirá que o usuário carregue seu programa em qualquer editor de texto que tenha a possibilidade de ler arquivos em ASCII como os programas Magic Window II, Appleworks,etc, por exemplo. Dessa forma o usuário poderá utilizar todas as facilidades do editor de textos para realizar alterações e complementações em seu programa.

Ao fim das mudanças no programa, utilizando-se do editor de texto, o usuário deverá gravar o mesmo também em código ASCII. Isso feito, o usuário poderá "carregar" o arquivo ASCII na memória do computador para, posteriormente, processá-lo (rodá-lo). Para tal, deve utilizar a instrução EXEC seguida do nome do arquivo.

Para operar a transformação (Applesoft X ASCII), utilize a seguinte linha de programa:

```
0 PRINT CHR$(4);"OPEN X": PRINT CHR$(4);
"WRITE X": POKE 33,33: LIST: PRINT CHR$(4);
"CLOSE X": END
```

Essa linha deve ser a primeira em seu programa. Após digitá-la, processe o programa (RUN). Observe que a única linha a ser processada será a descrita acima, de número zero, pois ao final desta temos a instrução END, que sinalizará o fim do programa. Isso feito, seu programa em Applesoft já será um arquivo tipo texto, podendo ser lido pelo editor de texto. Ao carregar o mesmo em um editor de texto, essa linha zero fará parte integrante do programa, portanto, não esqueça de removê-la.

---

11) Rotina de Localização de Endereço em Binário:

Muitas vezes o usuário desejará carregar na memória do computador um programa em forma Binária (BLOAD) para, posteriormente, transportá-lo para outro disco, regravando-o (BSAVE). Para tal operação torna-se necessário saber o comprimento deste programa em Linguagem Máquina.

Para conhecê-lo utilize a sub-rotina abaixo.

```
10  HOME
20 D$ = CHR$ (4)
30  INPUT "ENTRE O NOME DO ARQUIVO BINARIO:
";N$
40  PRINT
50  PRINT "AGUARDE !"
60  PRINT
70  PRINT D$;"BLOAD";N$
80  PRINT
90  PRINT N$;",A"; PEEK (43634) + PEEK
(43635) * 256;",L"; PEEK (43616) +  PEEK
(43617) * 256
```

O programa acima solicitará o nome do arquivo em Linguagem Máquina. Após o usuário entrar com o nome, o programa o carregará na memória e calculará seu comprimento, apresentando-o no monitor do vídeo.

APÊNDICE "A"

CÓDIGOS DE ERRO EM APPLESOFT E D.O.S.

0	NEXT sem FOR
1	Linguagem não disponível
2 ou 3	Erro de parâmetro
4	Protegido contra escrita
5	Fim de dados
6	Arquivo não encontrado
7	Volume não coincidente
8	Erro de Entrada/Saída
9	Disco cheio
10	Arquivo travado
11	Erro de Sintaxe
12	Não há Buffers disponíveis
13	Tipo de arquivo não coincidente
14	Programa muito comprido
15	Comando não direto
16	Erro de Sintaxe
22	RETURN sem GOSUB
42	Fora do DATA
53	Quantidade ilegal
69	Número fora do limite
77	Limite de memória ultrapassado
90	Declaração indefinida
107	Subscrito-erro de DIMensionanento
120	Arranjo redimensionado
133	Divisão por zero
163	Tipo não combina
176	Concatenação muito longa
191	Mais de dois IF...THEN...
224	Função indefinida
254	Resposta errada ao INPUT
255	CTRL-C pressionado

Segue uma breve explicação de cada Mensagem de Erro citada na página anterior. Após cada explicação do erro informamos se este ocorreu em Applesoft ou em DOS.

Código 0 (NEXT sem FOR):

Ocorre quando se tenta executar um loop dentro de um programa. Tal operação exige as instruções NEXT em conjunto com a instrução FOR. Caso uma delas não esteja no programa, ocorrerá um erro.
Instrução do Applesoft.

Código 1 (linguagem não disponível):

Ocorre quando a linguagem requerida não está disponível em memória.
A maioria dos Sistemas Operacionais de Disco (Versão 3.3), ao serem carregados, verificam a existência de uma interface de expansão de 16 (dezesseis) KBytes inerentes à própria memória do computador ou em interfaces independentes. Caso seja encontrada, armazenará nesse espaço de memória o Integer Basic, que é uma versão antiga do Applesoft. Nesse caso, com o Integer Basic alojado, para utilizá-lo bastará digitar a instrução INT seguida da tecla RETURN. Para sair do Integer Basic e retornar ao Applesoft, bastará digitar a instrução FP seguida da tecla RETURN.
Instrução do DOS.

Código 2 ou 3 (erro de parâmetro):

Ocorre quando um determinado parâmetro não existe ou não está disponível, como, por exemplo, acionar um soquete (slot) de número 8, que não existe.
Instrução do DOS.

Código 4 (protegido contra escrita):

Ocorre quando se tenta gravar arquivo ou programa em um disco que contenha uma etiqueta adesiva em sua ranhura de gravação (acima, à direita). Para efetuar corretamente a operação de gravação, basta retirar tal etiqueta.
Instrução do DOS.

Código 5 (fim de dados):

Ocorre quando se tenta ler um arquivo do tipo texto além de seu final.
Instrução do DOS.

Código 6 (arquivo não encontrado):

Ocorre quando se tenta ler um arquivo do tipo texto, atribuindo-lhe um nome incorreto ou não existente.
Instrução do DOS.

Código 7 (volume não coincidente):

Ocorre quando não se determina corretamente o número do volume do disco a ser utilizado.
Instrução do DOS.

Código 8 (erro de entrada/saida):

Erro geralmente causado na manipulação de leitores de disco. Pode ocorrer ao se acessar um leitor de disco com a porta aberta ou sem disco, tentar acessar um disco não inicializado ou com defeito.
Instrução do DOS.

Código 9 (disco cheio):

Ocorre quando se tenta gravar programas ou arquivos em um disco que já não contenha mais espaço para tal operação. Nesse caso, utilize outro disco, inicializado para efetuar a mesma operação.
Instrução do DOS.

Código 10 (arquivo travado):

Ocorre quando se tenta gravar arquivo com o mesmo nome de outro já existente no disco, e que está travado. Para travar o arquivo utiliza-se a instrução LOCK. Pode ocorrer, também, quando se tenta deletar (apagar) um arquivo do disco, estando o mesmo travado. Nesse caso, para efetuar operação, basta destravar o arquivo usando a instrução UNLOCK.
Instrução do DOS.

Código 11 (erro de sintaxe):

Ocorre quando um erro de grafia (ou escrita) foi cometido durante a digitação do programa. Nesse caso, o computador paralisará a execução do programa, emitirá um

bip e indicará em que linha do programa ocorreu tal erro. Isso não acontecerá se a instrução ONERR GOTO estiver ativa.
Instrução do DOS.

Código 12 (não há buffers disponíveis):

Ocorre quando existem muitos arquivos abertos simultaneamente.
Instrução do DOS.

Código 13 (tipo de arq. não coincidente):

Ocorre quando se tenta processar um arquivo em binário, por exemplo, com a instrução RUN, quando o certo seria BRUN.
Instrução do DOS.

Código 14 (programa muito comprido):

Ocorre quando se tenta carregar um programa maior do que a memória que o computador pode armazenar.
Instrução do DOS.

Código 15 (comando não direto):

Ocorre quando se tenta processar instrução que somente pode ser executada dentro de um programa, como: GOTO, GOSUB e outras.
Instrução do DOS.

Código 16 (erro de sintaxe):

Ocorre quando um erro de grafia (ou escrita) foi cometido durante a digitação do programa. Nesse caso, o computador paralisará a execução do programa, emitirá um

bip e indicará em que linha do programa ocorreu o erro. Isso não ocorrerá se a instrução ONERR GOTO estiver ativa.
Instrução do Applesoft.

Código 22 (RETURN sem GOSUB):

Ocorre quando se tenta executar um desvio para uma sub-rotina dentro de um programa. Tal operação exige a instrução GOSUB em conjunto com a instrução RETURN (para retornar à instrução imediatamente posterior ao GOSUB). Caso uma delas não esteja no programa, ocorrerá um erro.
Instrução do Applesoft.

Código 42 (fora de DATA):

Ocorre quando se tenta ler uma tabela de DATAs, sendo que a mesma chegou ao seu final.
Instrução do Applesoft.

Código 53 (quantidade ilegal):

Ocorre quando se utiliza um número maior do que o possível, em tabuladores, por exemplo.
Instrução do Applesoft.

Código 69 (número fora do limite):

Ocorre quando foi utilizado um número muito grande ou muito pequeno.
Instrução do Applesoft.

Código 77 (limite de memória ultrapassado):

Ocorre quando se tenta carregar um programa maior do que a memória do computador pode armazenar.
Instrução do Applesoft.

Código 90 (declaração indefinida):

Ocorre quando o programa encontra uma ordem de desvio (GOTO, por exemplo) para uma linha não existente.
Instrução do Applesoft.

Código 107 (erro de dimensionamento):

Ocorre quando uma matriz foi superdimensionada, devendo-se diminuir o número de matrizes.
Instrução do Applesoft.

Código 120 (arranjo redimensionado):

Ocorre quando é solicitado o dimensionamento de um arranjo anteriormente dimensionado.
Instrução do Applesoft.

Código 133 (divisão por zero):

Ocorre quando uma operação de divisão tiver como seu divisor o número zero.
Instrução do Applesoft.

Código 163 (tipo não combina):

Ocorre quando se tenta usar uma variável numérica como se fosse alfanumérica.
Instrução do Applesoft.

Código 176 (concatenação muito longa):

Ocorre quando se tenta concatenar uma string com mais de 255 caracteres.
Instrução do Applesoft.

Código 191 (mais de dois IF...THEN..):

Ocorre quando mais de dois IF..... THEN... estiverem na mesma linha.
Instrução do Applesoft.

Código 224 (função indefinida):

Ocorre quando se tenta utilizar uma função não definida.
Instrução do Applesoft.

Código 254 (resposta errada ao INPUT):

Acontece quando é dada uma resposta errada a uma solicitação via teclado para a instrução INPUT. Por exemplo: o INPUT solicita uma entrada de dado no modo numérico e se responde com uma entrada de dado no modo alfanumérico.
Instrução do Applesoft.

Código 255 (CTRL-C pressionado):

Essa mensagem indica que as teclas CTRL e C foram pressionadas em conjunto.
Instrução do Applesoft.

APÊNDICE "B":

CÓDIGOS DE ERRO EM INTEGER BASIC

*** > 255 ERR

- o valor numérico deveria estar entre 0 e 255.

*** > 32767 ERR

- foi calculado um número maior que 3276 ou menor do que -32767

*** 16 FORS ERR

- mais de 16 Loops ativados.

*** 16 GOSUB ERR

- mais de 16 GOSUBs ativados.

*** BAD BRANCH ERR

- desvio para uma linha que não existe.

*** BAD NEXT ERR

- foi encontrado um NEXT sem seu respectivo FOR.

*** BAD RETURN ERR

- foram encontrados mais RETURNs do que GOSUBs.

*** DIM ERR
- dimensionado o mesmo arranjo duas vezes.

*** MEM FULL ERR

- fim da memória disponível.

*** NO END ERR

- o programa não foi terminado com END.

*** RANGE ERR

- foram utilizados valores fora dos limites da sintaxe.

RETYPE LINE

- reposta errada ao INPUT.

*** STRING ERR

- operação errada ou ilegal com concatenação de strings.

*** STR OVFL ERR

- o limite de caracteres por string foi ultrapassado.

*** SINTAX ERR

- erro de digitação na confecção do programa.

*** TOO LONG ERR

- fórmula muito longa.

APÊNDICE "C":

MAPA DE MEMÓRIA DOS COMPUTADORES APPLE II PLUS COM 48 KBYTES DE RAM

48K Bytes RAM	SISTEMAS DE PROGRAMAS, STACK E BUFFER DO TECLADO		0 767
	DISPONÍVEL PARA USUÁRIO ARMAZENAR TABELAS OU PROGRAMAS EM ASSEMBLER		768 1007
	ENDEREÇOS DO MONITOR		1008 1023
	PÁGINA 1, TEXTO E BAIXA RESOLUÇÃO (1024) BYTES)		1024 2047
	INÍCIO DOS PROGRAMAS EM BASIC	PÁG.2,TEXTO BAIXA RESOLUÇÃO (1024 BYTES)	2048 3071
			3072 8191
	PÁGINA 1, ALTA-RESOLUÇÃO (8192 BYTES)		8192 16383

Cont.

48K Bytes RAM	PÁGINA 2, ALTA-RESOLUÇÃO (8192 BYTES)	16384 24575
	(*)	24576 38399
	USADO PELO D.O.S. (10752 BYTES)	38400 49151
4K BYTES E/S	ENTRADA/SAÍDA, TECLADO, FALANTE, PADDLES, SLOTS (4096 BYTES)	49152 53327
12K BYTES	SISTEMA ROM INTERPRETADOR APPLESOFT E MONITOR	53248 65535

(*) - Acima da página 2 de alta-resolução existe, também, espaço disponível para armazenamento de programas e de variáveis. Para sistemas com expansões de memória, este espaço naturalmente aumentará.

APÊNDICE "D"

POSIÇÕES DE MEMÓRIA USADAS PELA LINGUAGEM APPLESOFT

MEMÓRIA	ENDEREÇO
INÍCIO DO PROGRAMA EM BASIC	2048
VARIÁVEIS	
VARIÁVEIS NUMÉRICAS, ALFANUMÉRICAS E ARRANJOS.	8191
PAG. 1 ALTA RESOLUÇÃO (GRÁFICOS)	8192 16383
PÁG. 2 ALTA RESOLUÇÃO (GRÁFICOS)	16384 24575
STRINGS	24576 38399

APÊNDICE "E"

CÓDIGOS DE CARACTERES ASCII

ASCII	CARAC.	ASCII	CARAC.
0	CTRL-@	31	N.E.
1	CTRL-A	32	ESPAÇO
2	CTRL-B	33	!
3	CTRL-C	34	"
4	CTRL-D	35	#
5	CTRL-E	36	$
6	CTRL-F	37	%
7	CTRL-G	38	&
8	CTRL-H ou <-	39	'
9	CTRL-I	40	(
10	CTRL-J	41	)
11	CTRL-K	42	*
12	CTRL-L	43	+
13	CTRL-M	44	,
14	CTRL-N	45	-
15	CTRL-O	46	.
16	CTRL-P	47	/
17	CTRL-Q	48	0
18	CTRL-R	49	1
19	CTRL-S	50	2
20	CTRL-T	51	3
21	CTRL-U ou ->	52	4
22	CTRL-V	53	5
23	CTRL-W	54	6
24	CTRL-X	55	7
25	CTRL-Y	56	8
26	CTRL-Z	57	9
27	ESC	58	:
28	N.E.	59	;
29	CTRL-SHIFT-M	60	<
30	CTRL-^	61	=

CÓDIGOS DE CARACTERES ASCII (cont...)

ASCII	CARAC.	ASCII	CARAC.
62	>	95	_
63	?	96	`
64	@	97	a
65	A	98	b
66	B	99	c
67	C	100	d
68	D	101	e
69	E	102	f
70	F	103	g
71	G	104	h
72	H	105	i
73	I	106	j
74	J	107	k
75	K	108	l
76	L	109	m
77	M	110	n
78	N	111	o
79	O	112	p
80	P	113	q
81	Q	114	r
82	R	115	s
83	S	116	t
84	T	117	u
85	U	118	v
86	V	119	w
87	W	120	x
88	X	121	y
89	Y	122	z
90	Z	123	{
91	[	124	|
92	\	125	}
93	]	126	~
94	^	127	DEL

APÊNDICE "F"

CÓDIGO DOS COMANDOS DO APPLESOFT

ABS	212	AND	205	ASC	230
AT	197	ATN	225	CALL	140
CHR$	231	CLEAR	189	COLOR	160
CONT	187	COS	222	DATA	131
DEF	184	DEL	133	DIM	134
END	128	EXP	221	FLASH	159
FN	194	FOR	129	FRE	214
GET	190	GOSUB	176	GOTO	171
GR	136	HCOLOR	146	HGR	145
HGR2	144	HIMEM	163	HLIN	142
HOME	151	HPLOT	147	HTAB	150
IF	173	IN#	139	INPUT	132
INT	211	INVERSE	158	LEFT$	232
LEN	227	LET	170	LIST	188
LOAD	182	LOG	220	LOMEM	164
MID$	234	NEW	191	NEXT	130
NORMAL	157	NOT	198	NOTRACE	156
ON	180	ONERR	165	OR	206
PDL	216	PEEK	226	PLOT	141
POKE	185	POP	161	POS	217
PRINT	186	PR#	138	READ	135
RECALL	167	REM	178	RESTORE	174
RESUME	166	RETURN	177	RIGHT$	233
RND	219	ROT	152	RUN	172
SAVE	183	SCALE	153	SCRN	215
SGN	210	SHLOAD	154	SIN	223
SPC	195	SPEED	169	SQR	218
STEP	199	STOP	179	STORE	168
STR$	228	TAB	192	TAN	224
TEXT	137	THEN	196	TO	193
TRACE	155	USR	213	VAL	229
VLIN	143	VTAB	162	WAIT	181
XDRAW	149				

A tabela anterior informa as palavras reservadas do Applesoft e seus respectivos valores em modo decimal. Todo programa descrito em Applesoft pode ser demonstrado (listado) em Linguagem Máquina, na tela do monitor.

Considere o seguinte programa em Applesoft:

```
10 HOME
20 PRINT "ABCD"
30 END
```

Para saber em que posição de memória se encontra o início do programa acima, utilize a seguinte instrução:

```
PRINT PEEK (104) * 256 + PEEK (103)
```

Como resposta à instrução acima, temos o número 2049. Tal número está apresentado em modo decimal e indica a posição de memória onde o mesmo se inicia. Porém, essa posição deve ser tranformada para o modo hexadecimal. Para executar esta transformação, utilize o programa de conversão listado neste livro no capítulo de sub-rotinas, ítem V.9.

Utilizado esse programa, obteremos a conversão do número decimal 2049 para seu correspondente em hexadecimal que é 0801. Esse último número representa, efetivamente, a posição inicial de memória onde o programa está armazenado.

Para listá-lo em Linguagem Máquina, siga o procedimento descrito a seguir:

Digite CALL -151, seguido da tecla RETURN. Esta instrução permite o acesso ao modo Monitor, onde poderemos trabalhar em Linguagem Máquina. Após esse procedimento, obteremos um "*" que é o sinal de prontidão dessa linguagem.

Para listar o programa feito em Applesoft já em Linguagem Máquina, digite o seguinte:

```
*0801.0810  (e tecle RETURN)
```

O asterisco já estará sendo apresentado no vídeo. O número 0801 é a posição inicial do programa na memória do computador. O ponto indica que o bloco a ser listado deve compreender a posição 0801 até a posição 0810. Esse último número (0810) representa a última posição de memória utilizada pelo programa.

A apresentação do programa em Linguagem Máquina seria assim:

```
0801- 07 08 0A 00 97 00 13
0808- 08 14 00 BA 22 41 42 43
0810- 44 22 00 19 08 1E 00 80
```

Muitas vezes o usuário se depara com um quadro de instruções semelhante ao acima, sem, contudo, conseguir entendê-lo. Para análise do mesmo vamos separar alguns códigos, convertê-los para o modo decimal (com o programa de conversão descrito neste livro) e, de posse desse último valor, procuraremos a instrução correspondente na tabela de palavras reservadas do Applesoft, bem como na tabela de códigos ASCII. Enfim, teremos:

CÓD HEXA	CÓD DEC	INSTRUÇÃO APPLESOFT	TABELA PESQUISADA
97	151	HOME	T.P.R.
BA	186	PRINT	T.P.R.
22	34	"	ASCII
41	65	A	ASCII
42	66	B	ASCII
43	67	C	ASCII
44	68	D	ASCII
22	34	"	ASCII
80	128	END	T.P.R.

Legenda:

COD HEXA:	códigos em hexadecimal
COD DEC :	códigos em decimal
INSTRUÇÃO APPLESOFT:	instrução do Applesoft
TABELA PESQUISADA:	tabela em que foi feita a pesquisa do código em decimal
T.P.R. :	tabela de palavras reservadas do Applesoft
ASCII :	American Standard Code for Information Interchange

Observe que temos acima, na coluna de instruções do Applesoft, o mesmo programa apresentado como exemplo.

Os outros códigos apresentados, e não selecionados para análise, representam informações de controle da própria Linguagem de Máquina, o que foge ao objetivo de nosso trabalho.

APÊNDICE "G"

CUIDADOS COM O MICRO E COM OS DISCOS

Neste apêndice estão apontados os cuidados que devemos ter com os microcomputadores e os discos flexíveis. Recomenda-se seguir 'com seriedade' as instruções abaixo, pois assim serão evitados riscos de diversos tipos ao equipamento e aos discos, bem como será obtido um prolongamento da vida útil de ambos.

A- Cuidados com o micro:

1. Evitar oscilações bruscas de temperatura;
2. Evitar umidade;
3. Evitar poeira;
4. Não tapar os orifícios de ventilação;
5. Não expor o equipamento a fontes magnéticas;
6. Não limpar o equipamento com substâncias líquidas. Utilizar 'pasta limpadora' própria para equipamentos eletrônicos;
7. Não colocar outros equipamentos de grande peso sobre o computador;
8. Colocar o aparelho longe de quaisquer outros aparelhos elétricos que possam provocar interferência, tais como fotocopiadora, telex, máquinas de escrever eletrônicas, aparelhos musicais, etc;
9. Cobrir o equipamento com capa apropriada e manter as portas dos leitores de disco fechadas quando o computador não estiver sendo usado.

B- Cuidados com os discos:

1. Manter os discos afastados ao menos 30 centímetros de qualquer campo eletromagnético, como monitores, impressoras, fotocopiadoras, máquinas de escrever eletrônicas, aparelhos musicais e etc;
2. Nunca tocar a superfície descoberta do disco;
3. Inserir cuidadosamente os discos no disk drive;
4. Guardá-los na capa após o uso;
5. Nunca dobrá-los;
6. Mantê-los entre 10 e 50 graus Celsius;
7. Ao etiquetá-los, primeiro escrever na etiqueta e depois colar;
8. Protegê-los da poeira e do sol;
9. Manter um sistema de 3 cópias de discos de dados (backups).

índice Alfabético Remissivo

CALL.....11
CALL -151.....13
CALL -155.....13
CALL -167.....14
CALL -198.....14
CALL -211.....14
CALL -380.....14
CALL -384.....14
CALL -676.....15
CALL -678.....15
CALL -756.....16
CALL -856.....17
CALL -868.....18
CALL -912.....18
CALL -922.....19
CALL -936.....19
CALL -958.....19
CALL -998.....20
CALL -1008.....20
CALL -1036.....21
CALL -1052.....21
CALL -1184.....21
CALL -1216.....22
CALL -1233.....22
CALL -1321.....22
CALL -1953.....23
CALL -1994.....23
CALL -3082.....24
CALL -3086.....24
CALL 42350.....24
CALL 62450.....25
CALL 62454.....25
Código de Caracteres ASCII.....92
Cuidados com o Micro.....98
Cuidados com o Disco Flexível.....99

índice Remissivo (Cont.)

Mapa de Memória (apple II com 48K).....89
PEEK.....10
PEEK -16286/-16287.....26
PEEK -16303 ou POKE -16303,0.....27
PEEK -16304 ou POKE -16304,0.....27
PEEK -16336.....27
PEEK -16384.....28
PEEK 44.....29
PEEK 78.....30
PEEK 103/104.....30
PEEK 105/106.....31
PEEK 111/112.....31
PEEK 117/118.....31
PEEK 119/120.....32
PEEK 123/124.....32
PEEK 129/130.....33
PEEK 175/176.....34
PEEK 202/203.....34
PEEK 218.....35
PEEK 222.....35
PEEK 224/226.....37
PEEK 241.....37
PEEK 978.....38
PEEK 2040.....38
PEEK 43624.....38
PEEK 43626.....38
PEEK 43697.....38
PEEK 44033.....38
PEEK 46017.....39
PEEK 46064.....39
POKE.....10
POKE -16302,0.....64
POKE 32,10.....40
POKE 33,16.....41
POKE 34,10.....42
POKE 35,10.....43
POKE 36,X.....44

POKE 37,X.....45
POKE 48,X.....45
POKE 50,140.....46
POKE 50,X.....47
POKE 214,X.....48
POKE 216,X.....48
POKE 241,256-X.....49
POKE 243,32.....49
POKE 243,128.....49
POKE 245,0/242,128.....50
POKE 1009,76/1010,241/1011,165/1012,166..50
POKE 1010,30/1011,60/1012,90.....51
POKE 1010,102/1011,213/1012,112.....51
POKE 1013,76/1014,88 /1015,252.....52
POKE 1013,76/1014,102/1015,213.....51
POKE 1013,76/1014,105/1015,255.....52
POKE 1013,76/1014,165/1015,214.....52
POKE 1013,76/1014,228/1015,251.....52
POKE 1204,X/POKE 2039,X.....53
POKE 2049,1.....54
POKE 40286,35/40287,216.....54
POKE 42768,234/42769,234/42770,234.....55
POKE 44033,X.....55
POKE 44452,24/44605,23.....56
POKE 44452,255/44513,67.....62
POKE 44505,234/44506,234.....57
POKE 44567,X.....57
POKE 44578,234/44579,234/44580,234.....57
POKE 44596,234/44597,234/44598,234.....58
POKE 44599,234/44600,234.....58
POKE 44611,X.....59
POKE 45991,X.....59
POKE 45992,X.....60
POKE 45993,X.....60
POKE 45994,X.....61
POKE X,Y (X entre 45999 e 46010).....61
Posições de Memória do Applesoft.....91

Programa de conversão Hexa<=>Decimal.....73
Relação Caracter X Código ASCII.....73
Rotina Hidden.....72
Sub-rotinas.....65
Sub-rotina Applesoft X Arquivo Texto.....76
Sub-rotina Gerando de Sons.....66
Sub-rotina Localizadora de End.Binário...77
Sub-rotina Marca Personalizada.....65
Tabela Comandos do D.O.S. 3.3 68
Tabela Mensagens de Erro.....70
Tabela de Instruções do Applesoft.....67
Tabela do Catálogo.....71
Tabela dos Cód.de Erro Applesoft/DOS.....79
Tabela dos Cód.de Erro Integer Basic.....87



Bibliografia Recomendada:

* Programming with BASIC Schaum's outline series in computers
  Byron S. Gottfries, McGraw-Hill

* Introducion al processamiento de datos para los negocios
  Orilia, McGraw-Hill

* Guia do Programador D.O.S.
  Roberto Massaru Watanabe
  Editora Aleph

* Assembly 6502
  Bernhard Wolfgang Schon
  Editora Aleph

* Guia do Programador CP/M
  Roberto Massaru Watanabe
  Editora Aleph

* What's Where in the APPLE
  William F. Luebbert
  MICRO INK

* DOS 3.3
  Sistema Operacional de Disco
  UNITRON

* MANUAL DE BASIC
  Herbert Peckham
  McGraw-Hill

* TK3000 IIe
  Manual de Operação
  MICRODIGITAL

OUTRAS PUBLICAÇÕES DA ALEPH

para linha APPLE

Editora Aleph

TABELA DE MNEMÔNICOS DO 6502

Bernard W.Schon

Tabela plastificada de consulta rápida com todos os Mnemônicos utilizados pelo microprocessador 6502 utilizado pelos computadores compatíveis com o APPLE II, indispensável para programadores em Linguagem de Máquina.

GUIA DO PROGRAMADOR D.O.S.

Roberto Massaru Watanabe

Como funciona e como utilizar o Sistema Operacional de Disco D.O.S. 3.3 usado pelos computadores da linha APPLE. Todos os segredos dos disquetes desvendados de maneira didática e simples.

GUIA DO PROGRAMADOR CP/M

Roberto Massaru Watanabe

Os segredos do Sistema Operacional mais difundido no mundo da computação: o CP/M. De que maneira o CP/M pode ser explorado pelos programadores em APPLE para melhorar a eficiência de um programa? Como agrupar rotinas em Linguagem de Máquina do 6502, do 8080 e do Z80 num único programa? Como chamar rotinas do CP/M a partir de um programa em BASIC ? O CP/M do APPLE é diferente dos outros CP/M ? Com a entrada do Z80, o 6502 perde sua função ?
Estas e muitas outras perguntas podem ser respondidas com a leitura deste guia que também contém a listagem disassemblada do CP/M para APPLE.

ASSEMBLY 6502

Bernard W.Schon

Neste livro são abordados, de maneira clara e didática, os segredos da Linguagem de Máquina do microprocessador 6502, cujo conhecimento é indispensável ao programador que não quer se sujeitar às limitações da linguagem BASIC.
Trata-se de uma obra pioneira, escrita especificamente para o público brasileiro e que leva em conta a existência de computadores nacionais compatíveis com o APPLE norte-americano.

para linha SINCLAIR

INFORMÁTICA NA ESCOLA

David Zumerkorn & Osmir Foltran

Livro didático fartamente ilustrado para introduzir a criança e o jovem no maravilhoso mundo da computação.
Iniciação à linguagem BASIC do TK-85 (e compatíveis SINCLAIR) desenvolvida de maneira clara e didática.

BASIC TK

Pierluigi Piazzi & Flavio Rossini

Um verdadeiro curso de BASIC desenvolvido ao longo de 3 volumes, começando pela iniciação do usuário leigo e chegando até os detalhes mais técnicos dos micros da linha SINCLAIR. Obra didática e clara, escrita por dois professores e indispensável para quem quer esgotar o assunto.

EVOLUINDO NO BASIC TK

Bernardo Stein

Num diálogo bem humorado com o leitor, Stein descreve como construir 16 programas interessantes para micros da linha SINCLAIR. Além de obter um software interessantíssimo o leitor adquire todo "know how" necessário para melhorar e personalizar seus próprios programas.

JOGOS EM LINGUAGEM DE MÁQUINA

Piazzi et al.

Obra dividida em 3 volumes nos quais são fornecidos inúmeros programas em Assembly para micros da linha SINCLAIR. Uma maneira divertida e instrutiva de se montar uma completa biblioteca de software.

USANDO LINGUAGEM DE MÁQUINA

Mario Schaeffer

Uma obra original e didática para aprendizado e consulta. Muitas dicas para o uso da linguagem de máquina do Z80 em computadores compatíveis com os micros SINCLAIR.
Sugestões para o uso "balanceado" da LM e do BASIC. Como usar sub-rotinas da ROM inclusive para cálculos científicos.
Muitos programas disassemblados e comentados linha por linha, inclusive de inteligência artificial.

DISSECANDO JOGOS

Carlos Eduardo Rocha Salvato

Os sete jogos contidos neste livro têm uma característica única, o leitor participa do processo de criação acompanhando a montagem linha por linha.
Modificações e aperfeiçoamentos são propostos e resolvidos com detalhes.

CRIANDO EM LINGUAGEM DE MÁQUINA

Samuel Ejchel

Não basta conhecer as instruções do ASSEMBLY para se criar um programa interessante: existem truques e "dicas" indispensáveis para um bom programador.
Gerenciando a tela, utilizando o teclado, usando o "joystick", gerando movimentos múltiplos, criando opções, implementando requintes.

O SEU MICRO E O MUNDO EXTERNO

Bernhard Wolfgang Schon

Neste livro, através de projetos claros e utilizando componentes facilmente encontrados no mercado nacional, o autor demonstra como é fácil acessar o "mundo externo" do microcomputador. Partindo de um simples circuito básico, o "hobista" pode montar os mais variados periféricos para micros da linha SINCLAIR e, com algumas adaptações, APPLE.

para linha MSX

APROFUNDANDO-SE NO MSX

Piazzi, Maldonado, Oliveira et al.

Para quem quer conhecer todos os detalhes da máquina: como usar os 32kb de RAM escondidos pela ROM, como redefinir caracteres, como usar o SOUND, como tirar cópias de telas gráficas na impressora, como destravar fitas. Todos os detalhes da arquitetura do MSX, o BIOS e as variáveis do sistema comentadas e um poderoso programa Disassembler.

TABELA DE MNEMÔNICOS Z80

Cassiano Roda

Uma tabela com todos os Mnemônicos do microprocessador Z80 (o microprocesador do MSX) relacionados com seus códigos hexadecimais. Indispensável para quem está começando a programar em linguagem de máquina e não dispõe de um programa Assembler.

USANDO O DISK DRIVE NO MSX

Rubens Pereira Silva Jr.

O DISK BASIC, o MSX DOS e o CP/M para o MSX comentados detalhadamente. Desde a instalação até os usos mais sofisticados. Completando a obra, um dicionário de todos os comandos e funções desses três sistemas operacionais. Indispensável para quem tem um drive!

SISTEMA DE DISCO PARA MSX

Oliveira e Pereira

Como instalar a interface CDX-2 utilizando o sistema operacional residente em sua ROM, o poderoso BASIC DE DISCO. Como tirar o máximo proveito do SOLX-DOS, sistema operacional totalmente compatível com o MSX DOS e semi-compatível com o CP/M. Como converter programas instalados em CP-500 e em Sistema 700 para o MSX e vice-versa. Complementando a obra, um detalhado dicionário de todos os comandos e funções, repleto de exemplos didáticos e aplicações. Uma obra importantíssima para quem quer tirar o máximo proveito de seu sistema MSX!

COLEÇÃO DE PROGRAMAS PARA MSX V.1

Oliveira et al.

Uma coletânea de programas para o usuário principiante em MSX. Jogos, músicas, desenhos e aplicativos úteis apresentados de modo simples e didático. Todos os programas têm instruções de digitação e uma análise detalhada, explicando praticamente linha por linha o seu funcionamento. Todos os programas foram testados e funcionam! A maneira mais fácil e divertida de entrar no maravilhoso mundo dos micros MSX.

COLEÇÃO DE PROGRAMAS PARA MSX V.2

Oliveira et al.

Programas com rotinas em BASIC e Linguagem de Máquina. Jogos de ação e inteligência, programas didáticos, programas profissionais de estatística, matemática financeira e desenho de perspectivas. Utilitários para uso da impressora e gravador cassete.

JOGOS DE HABILIDADE PARA MSX

Dias, Guazelli, Martins e Sung

Coletânea de jogos de ação e inteligência de nível profissional, usando sempre que necessário, os recursos da Linguagem de Máquina. Asteróides, Coelho Maluco, Alcatraz Gincana, etc, são alguns programas listados neste livro junto com as instruções sobre como utilizá-los.

LINGUAGEM BASIC MSX

Denise Santoro Cruz

Uma verdadeira enciclopédia do BASIC MSX onde todos os comandos e funções do BASIC residente são listados em ordem alfabética. Para cada comando é dada a explicação do que ele faz, sua sintaxe e um exemplo elu-

cidativo. Obra de consulta, indispensável para quem programa em BASIC MSX, adotada pela GRADIENTE para acompanhar o microcomputador Expert.

HOTLOGO-primeiro passos

Godoy, Lacerda, Lepíscopo e Mendes

Livro dedicado às crianÇas que começam a entrar no mundo maravilhoso da informática. A linguagem utilizada é o LOGO para MSX, disponível em cartuchos, uma das versões mais completas da linguagem desenvolvida por Papert, com base no estudo do desenvolvimento da criança realizado por Piaget.

HOTDATA

Roberto Massaru Watanabe

O gerenciamento de um banco de dados explicado de maneira didática para quem utiliza um cartucho HOTDATA no MSX.

HOTWORD

Roberto Massaru Watanabe

O processamento de textos no micro, explicado de maneira didática e simples para quem usa um cartucho HOTWORD no MSX.

HOTPLAN

Roberto Massaru Watanabe

Uma das melhores planilhas eletrônicas, explicada com exemplos e figuras para os que utilizam o cartucho HOTPLAN no HOTBIT.

DOMINANDO O EXPERT

Denise Santoro Cruz

Os primeiros passos na programação e utilizaÇão do microcomputador Expert com explicaÇões didáticas para os usuários principiantes. Obra adotada pela GRADIENTE.

RESUMO DE OPERAÇOES DO EXPERT

Renato da Silva Oliveira

Tabela de consulta rápida com todos os comandos e funções do BASIC MSX. Complemento extremamente útil da obra "LINGUAGEM BASIC MSX" para ser deixado constantemente ao lado do teclado de seu MSX.

PROGRAMAÇAO AVANÇADA EM MSX

Maldonado, Figueredo e Rossetto

Neste livro todos os segredos da arquitetura dos micros MSX são comentados e os truques e macetes para controlá-los através da Linguagem de Máquina do Z80 são exaustivamente ensinados.

Para receber mais informações sobre nossos livrô e ganhar uma assinatura gratuita de nosso boletim informativo, envie seu nome e endereço completos (inclusive CEP !!) para:

Editora Aleph
Boletim Informativo
CP 20 707
CEP 01498 S.Paulo SP

A composição deste livro foi feita numa impressora Mônica Plus 6030 da Elebra, utilizando-se fontes alternativas para APPLE.

Segredos do apple



A Coleção SEGREDOS DO APPLE da editora ALEPH, propõe-se desvendar os detalhes de um dos primeiros e mais disseminados microcomputadores do mundo. A eventual obsolescência pela qual deveria passar o APPLE, numa área de tão bruscas mudanças tecnológicas como é a computação, é constantemente adiada pela riqueza de periféricos e, principalmente, pela superabundância de software.

Neste volume são fornecidas mais de 100 instruções PEEK, POKE e CALL para agilizar a execução de programas, permitindo a velocidade da linguagem de máquina a trechos de programas em BASIC.

Além de permitir o aprofundamento na utilização de computadores da linha APPLE, esta obra representa uma valiosa fonte de consulta, indispensável a programadores e usuários principiantes que desejam implementar rotinas rápidas em seus programas.

Todas as rotinas e instruções deste livro foram testadas em um microcomputador compatível com APPLE II+ (UNITRON) e APPLE IIe (TK 3000), estando indicado, no texto, o(s) tipo(s) de computador no qual a instrução indicada funciona.

PEEK, POKE & CALL

Editora Aleph