



# DOS BOSS<sup>TM</sup> DISK COMMAND EDITOR

**DOS Boss** is an extremely easy-to-use, thoroughly-documented Apple utility package from Beagle Bros. that will customize your disk system and truly personalize your personal computer!

### Rename Apple's DOS Commands.

To make a change, simply run DOS BOSS, enter the original form of any DOS command (say "CATALOG"), then your new command word (say "CAT") and that's all! Now "CAT" will catalog your disks! Other changes are created with equal ease. For example...

### Rewrite Error Messages.

"SYNTAX ERROR" can be renamed "CANNOT COMPUTE" or "TRY AGAIN!"; "DISK FULL" can be "BUP!"; anything you want! Any unauthorised attempt to save a program can produce a (better!) "NOT COPYABLE" message.

### Run programs with one keystroke!

DOS BOSS completely simplifies Apple program selection. Now you will use a single letter next to each file name in your catalogs. Pressing the appropriate key will RUN, BULK or EXEC the chosen program for you automatically! No list to the number of file names!

### Customize your Catalogs!

Group your programs by file type (A, I, B & T) when you catalog. OR catalog only the file-types you want.

Replace Apple's unprecise "DOS VOLUME:" heading (our personal disk number, title or name, with or without the Volume Number.

Convert long catalogs to 2 or 4 columns so that EVERY file name fits on the screen. Date or other sector numbers and language codes too.

**DOS BOSS's change features may be appended to any or all of your programs, so that anyone using your disks (floppy or hard) on any Apple will be formatting DOS the way YOU designed it!**

### \* Plus The DOS BOSS BOOK!

Easy-to-Read Documentation and Juicy Apple Information!

You don't need to be an Apple Expert just to read our instructions. The **DOS BOSS BOOK** is an excellent-written Apple learning tool covering not only DOS BOSS but a whole range of Apple facts and features. **PART 1** discusses each DOS BOSS feature.

- **PART 2** is an extensive collection of instructions for executing a whole pile of DOS & non-DOS tricks, experiments and customizations, such as:
  - Make your program un-stable or un-killable;
  - (ignore and try to) create a strange new sort of Apple bug!

• Type inverse or flashing characters directly onto the screen. Put "beagle" catalog sub-headings into your catalogs, rename if you like and formatted the way you want!



by Bert Kenney and Jack Casady  
(1.9 or 3.3 AppleSoft Disk)



# The DOS BOSS BOOK

Complete Instructions for Using



by Bert Kersey and Jack Cassidy

**Plus a New Assortment of  
APPLE II TIPS AND TRICKS**

## TABLE OF CONTENTS

What's DOS? . . . . .	2
What's DOS BOSS? . . . . .	2
Using DOS BOSS. . . . .	3
(M) MENU. . . . .	3
(C) DOS COMMAND CHANGES . . . . .	3
(E) ERROR MESSAGE CHANGES . . . . .	6
(X) CATALOG FORMATTER . . . . .	8
(V) VOLUME HEADING CHANGES. . . . .	8
(F) FILE CODE CHANGES . . . . .	9
(S) SAVING DOS CHANGES. . . . .	10
Key-Cat . . . . .	11
Bait-Cat. . . . .	11
LP and NU . . . . .	11
Changing DOS Without DOS BOSS . . . . .	12
DOS & Non-DOS Tips and Tricks . . . . .	15
Cruising Through DOS. . . . .	23
Appendix . . . . .	25
DOS BOSS Program Listing. . . . .	25
ASCII Screen Characters Chart . . . . .	34
A Word From Our Sponsor . . . . .	36

Entire contents Copyright © 1981, Beagle Bros. Micro Software  
4315 Sierra Vista, San Diego, CA 92103, (714) 996-6400.

"Apple", "Apple II", and the Apple Logo are all  
registered trade marks of the Apple Computer Company.



## What's DOS?

DOS is Apple's "Disk Operating System". Without it, your Apple simply does not know how to perform any function that involves a disk-- load a program from disk, catalog, tell you "FILE NOT FOUND", access a text file, etc., etc. Let alone start the motor on your disk drive. Without DOS, your Apple DOES know how to execute BASIC functions (I assume you are using AppleSoft or Integer BASIC), because BASIC is BUILT IN to your Apple in the form of unchangeable hardware, or "ROM" (Read Only Memory). DOS (thruess with "boss", by the way) is actually a complex machine-language PROGRAM that is entered into your Apple's memory in "RAM" (Random Access Memory) each time you boot a disk. DOS normally remains in memory as long as your Apple is turned on and is not affected or changed by anything you ordinarily do-- programming, loading, saving, deleting, etc.

So, after you boot a disk (load DOS), your Apple knows TWO sets of instructions, BASIC and DOS. When you enter an instruction through the keyboard, the Apple checks it FIRST to see if it is a DOS command. THEN to see if it is a BASIC command. If you type "ABCFED" with a carriage return, for example, the Apple checks its entire 29 word DOS command vocabulary (words like "CATALOG", "INIT", "DELETE", etc.) to see if it knows "ABCFED". If it doesn't, it then checks its BASIC vocabulary (words like "LIST", "GOTO", "AND", "NEXT", "POKE", etc.). If it can't find "ABCFED" there, it gives up and prints "SYNTAX ERROR" or "### SYNTAX ERR". If it DOES know the word that you have typed, it executes the command according to the instructions that reside in memory, either DOS or BASIC, depending on where the command was found. All of the above takes approximately no time at all.

## What's DOS BOSS?

Since DOS is an ACCESSIBLE written program in RAM and not a permanent collection of hardware like BASIC, you can CHANGE it to suit your desires and to have more control over your computer. DOS BOSS is the key to making these changes. With DOS BOSS, you will have immediate access to DOS's most visible functions and features. With this book, you will have even further control, and learn a bit more about what goes on inside your Apple's "brain". I have written as much as possible from a beginner's viewpoint, assuming that you know nothing of machine level programming or the way a computer works. Technical details, whenever possible, have been omitted or written in English. Let's get on with it and have some fun!

## Using DOS BOSS

DOS BOSS is easy to use. Let's use it!

### STARTING OUT

Boot the DOS BOSS disk. You are now operating under normal DOS conditions with your normal Apple. Now type RUN DOS BOSS and hit "return".

### (M) MENU

In a few seconds, you will see the DOS BOSS MENU, showing the DOS change options available to you. To the left of each option is a letter inside of ( ) brackets. This indicates that only one keypress is needed to select the option. Most of what you do with DOS BOSS will be done with one keypress (no "return" necessary).

You will need to return to the MENU each time you want to select another DOS BOSS feature. To do so, simply type an M or an H + carriage return (cr) from almost any part of DOS BOSS. If you wanted to, you could even hit RESET and RUN DOS BOSS again without losing the 35 changes you had made so far. DOS BOSS's "variables" are actually memory views in DOS and will not be cleared when you RUN any program or change languages because DOS is tucked safely above HIMEM, the highest memory location accessible by your BASIC (AppleSoft or Integer) programs.

Let's cover the DOS BOSS features--

### (C) DOS COMMAND CHANGES

Select C from the MENU and you will see presented with Apple's 29 DOS COMMANDS, ready to be changed. To the right of each command should be the word "SAME". This means that each command is in its standard form (CATALOG will catalog LOAD will load, etc.). If all the commands are not marked "SAME", you may standardize them at any time by entering a \$ sign.

Let's change a command: Select "CATALOG" as a test command by pressing the R key (no need to press "return"). Now, type in a new command, seven characters or shorter (no spaces, commas or colons allowed in commands!), and hit "return". In a few seconds, you will see your new command in inverse to the right of CATALOG. During the pause between your inputs, the Apple is making room for the new command, inserting the new command in DOS, and "sliding" all other commands up or down the adjacent. See "CRUISING THROUGH DOS" later in this book for a memory layout of the DOS commands.

For a test, enter "CAT" as the new CATALOG command. Exit the COMMAND CHANGE MODE by pressing M for MENU and exit DOS BOSS by pressing Q for QUIT. Now try to CATALOG a disk. The word CATALOG gives you a SYNTAX ERROR (the Apple thinks it means CATALOG AL001), BUT the word CAT:atalogs your disk! Much easier to



type, right? How about changing the command to CC? Simpler yet! Type RUN now and change it.

There are 132 character locations set aside for DOS Commands. The total length of all 28 commands may not exceed 132 characters. DOS B0SS will let you lengthen a short command if you have shortened another and have the spare characters available. To lengthen the FP command to the word APPLESOFT, for example, you could shorten CATALOG to CAT and VERIFY to VER. You have created seven spare characters, and now you can lengthen FP to APPLESOFT. Each new command may be as long as the number of underscores (.....) shown before you enter the new word. DOS B0SS will not allow a command longer than nine characters. In DOS B0SS, on certain long commands, the new command will overwrite the original on the left. This was done in the interest of saving screen space.

## COMMAND POSSIBILITIES

You may want to change all or just a couple of commands to cover a given situation. Let's take a look at just a few change ideas. More command change ideas appear in the Error Message section of this book.

CATALOG-- Shorten it. See above.

INIT-- Accidentally typing INIT could prove disastrous! For me, it resembles "INT", the Integer command just a bit too much. Why not change INIT to KILL or a longer code word like FORNIGHT?

INT-- Another command you don't want to accidentally type, since it will ERASE any BASIC program in memory. If you have the spare characters, you could rename this command INTEGER or I/9 or 9.

FP-- Beginners have a hard time remembering this one. A/S or APPLESOFT or J (Shift-M) might make more sense to you.

EXEC-- I use EX. I have a Text File called LIST that turns on my printer, sets it up for 80 characters per line, lists a program, then turns the printer off. To list any program on my printer, I simply type EXLIST (meaning EXEC the file named LIST), and my printer goes to work with my program still intact! The BASIC LIST command, of course, still lists on the screen in the normal manner.

BRUN-- Typing HELP can BRUN a user instruction file named LP if you rename the BRUN command HE (HELP then means BRUN LP). Try it. There's a sample LP program on the DOS B0SS disk which runs the ASSISTANCE program.

VERIFY-- VV is perfect here! much shorter (and REAL hard to misspell!)

LOAD & SAVE-- Special commands! If you rename them, you must name other commands "LOAD" and "SAVE" or your system will freeze if they are accidentally typed, thinking you are doing a cassette LOAD or SAVE. More later in the Error Message Section.

RUN-- RUN by itself is a BASIC command. With anything following it other than a carriage return, it is a DOS command. It is best not to mess with DOS's RUN command... Oh, SO AHEAD! It's fun to confuse your Apple! If you rename RUN, R, for example, your Apple could misinterpret the BASIC command RUN as RUN UM. If things get totally out of hand while you're experimenting, you can always re-boot.

## WATCH-OUT-FORS

1. If you have changed a command (say CATALOG to CAT) and encounter a ctrl-D execution of that command in its REGULAR FORM in a program (like PRINT CHR\$(4);"CATALOG"), the program will bomb with a SYNTAX ERROR or SYNTAX ERROR. Boot normal DOS if you aren't familiar with a program.
2. BE CAREFUL WITH ONE-LETTER COMMANDS! It's best to use a letter that no other command starts with.
3. Giving a DOS command a BASIC command name will make the BASIC command unusable. Change CATALOG to LIST, for example, to make your programs un-listable. Remember, Apple checks DOS, THEN BASIC for commands.
4. Don't use spaces in commands. If you want a two-word command like CAT LIST, enter it in DOS B0SS as CATALOG. The Apple is trained to ignore space-> For example, the normal CATALOG command will execute fine as CAT A LOG or CATALOG.
5. Don't put commas or colons in commands. Apple's INPUT function won't allow them, so DOS B0SS won't either.
6. All DOS commands except CATALOG, CLOSE, INT & FP must be followed by other words or characters, usually a file name, to be valid. Just thought I'd mention it.
7. With duplicate DOS commands, only the first one will function.

## SELF-TEST

You can quickly print all DOS commands in their current form without having to enter the Command Change mode--

1. Quit DOS B0SS with a Q.
2. Type RUN 22222 ("return").

If you see some inverse '+', those are spare characters.

## (E) ERROR MESSAGE CHANGES

Error Message changes are made just like Command changes. Select the message and type in a new one. When you are finished, hit H for MENU. Changing error messages is faster, because LENGTHENING MESSAGES IS NOT ALLOWED, and DOS BOSS lengthens any shorter-than-standard message with invisible spaces. In DOS BOSS, on certain long error messages, your new message will overwrite the original on the left. This was allowed in the interest of saving screen space.

### TESTING ERROR MESSAGES

To see if your altered error messages are printing the way you want, you can trigger each with the following (standard) DOS commands:

LANGUAGE NOT AVAILABLE-- LOAD an INT file with an AppleSoft-only Apple.

RANGE ERROR-- MAYFILES 17.

WRITE PROTECTED-- SAVE FILE with a write-protected disk.

END OF DATA-- WRITE an empty text file, TFILE, and then EXEC TFILE.R2.

FILE NOT FOUND-- LOCK X (no program X on disk).

VOLUME MISMATCH-- LOCK X,V123 (123 is wrong volume number).

I/O ERROR-- CATALOG with no disk in drive.

DISK FULL-- SAVE X with a full disk.

FILE LOCKED-- SAVE FILE where FILE is locked.

SYNTAX ERROR-- CATALOGX.

NO BUFFERS AVAILABLE-- Set MAYFILES 1. Then OPEN a file, and CATALOG.

FILE TYPE MISMATCH-- BLOAD FILE where FILE is in AppleSoft or Integer.

PROGRAM TOO LARGE-- Lower HIMEM to 3000 and LOAD a large program.

## COMPUTERS-DON'T-HAVE-TO-TALK-LIKE-COMPUTERS DEPARTMENT

Come on gang! Loosen up your Apple! Life is too short for those D-U-L-L error messages!

DISK FULL-- Maybe make it BUBBI instead.

LANGUAGE NOT AVAILABLE-- How about NO SPEAKA DA INTEGER!

FILE NOT FOUND-- TRY AGAIN, BOZO!

PROGRAM TOO LARGE-- CANNOT COMPUTE!

ANYTHING-- How about BUMMER!

Or RATS!!!

Or CAN'T YOU SPELL?

Or HUH??

SYNTAX ERROR-- Should be renamed DOS ERROR or anything not resembling AppleSoft's SYNTAX ERROR or Integer's ## SYNTAX ERR. When a DOS error is encountered, you should instantly be able to recognize it as such.

## COMMAND/MESSAGE COMBOS:

This is fun! I'll give you a some examples, and you take it from there. These changes may all be appended to any program (see later in the "SAVING DOS CHANGES" section, page 10).

1. Change the SAVE command to KEEP.
2. Change the READ command to SAVE.
3. Change the NOT DIRECT COMMAND error message to NOT COPYABLE!!

Now when anyone tries an unauthorized SAVE of your prized STAR-SHOOTERS program, he types SAVE STAR-SHOOTERS and gets a "NOT COPYABLE!!" message. The Apple thinks "SAVE" means "READ" and prints the message you have assigned to an illegal direct READ command! YOU can still save the program by typing KEEP STAR-SHOOTERS.

See page 21 for more ideas on this.

Or...

1. Change the CATALOG command to CAT.
2. Change the VERIFY command to CATAL.
3. Change the FILE NOT FOUND message to ## TYPE "CAT".

Here, the Apple thinks CATALOG means VERIFY file 06, can't find file 06, and prints the instructions for your new CATALOG command! Handy if a stranger is using your DOS BOSSed Apple and wants to catalog.

Or...

1. Change the READ command to LIST.
  2. Change the NOT DIRECT COMMAND message to NOT LISTABLE!
- Another in a long series of ways to Non-Justify your programs to frustrate beaky people!

## SELF-TEST

You can quickly print all DOS error messages in their current form without entering the Error Message Change mode--

1. Quit DOS BOSS with a Q.
2. Type RUN 22333 ("return").



### (X) CATALOG FORMATTER

It can be frustrating when all of your file names won't show on the screen at one time. DOS BOSS's Catalog Formatter can increase this to 88. Enter this mode from the MENU with an X. You can re-layout your catalogs as indicated by the chart on the screen. Try all of the options. Each selection will instantly show you your new catalog format, so BE SURE YOU HAVE A DISK IN YOUR DRIVE. Options 3 & 4 will eliminate your language codes (A,I,B & T) and sector numbers for added horizontal space. These codes are usually not necessary anyway.

One drawback to the two and four-column catalogs is that file names longer than the maxiums shown on the right of the screen will be chopped off when presented. You must either shorten these file names (the best solution) or REMEMBER them so you can access them by their real titles.

DOS BOSS creates these multi-columned catalogs by simply removing the carriage returns after each file name. Three-columned catalogs are not possible using this procedure, since an odd number of characters would be required on each line, and it takes 40 characters horizontally to fill the screen. See "Making Changes Without DOS BOSS" a few pages from here for more.

Don't use file names with hidden control characters with the split catalogs! the columns will get out of alignment. Two and four-column catalogs will sometimes look better if you make the LOCK and UNLOCK codes INVERSE with DOS BOSS. See <F> FILE CODE CHANGE.

### (Y) VOLUME HEADING CHANGES

To see Apple's "DISK VOLUME 254" message is clunky, and takes up too much screen space. DOS BOSS will permit you to replace the heading with anything you want, up TO 16 CHARACTERS IN LENGTH. You can also eliminate DR include the Volume Number itself. Select V from the MENU for a test run.

A sample of the current heading will be displayed on the screen. To remove or include the Volume Number, press #. To change the message, press C, and type in a new heading followed by "return". "Return" alone will restore the standard (even if it IS clunky!) DISK VOLUME heading. You may also select N, I, or F for a Normal, Inverse, or Flashing heading. The Volume Number, when visible, must always be Normal.

Special Characters may be used to create trick titles. The @ will print as a carriage return for multi-line titles. The ^ will do a line feed. And the > will let you indent a title, nice in Inverse (normally, the Apple ignores a leading space on an INPUT; the > remedies this problem!). The < acts as a backspace. Fool around with these special characters, and you'll see the results. ALL CATALOGS WILL HAVE THIS TITLE until another disk is booted, or until a new title has been injected into DOS.

### DISK VOLTRIX

1. Make your heading GEORGE'S DISK??- and you will get a nice two-line heading with a V-254 (or whatever Volume Number the disk was INITIATED with) on the second line.

2. Try A^@^C^D^E^F in Inverse with no Volume Number! I like to use >BEAULEZ>BROS.> in Inverse.

3. Or HA 002 HELLO with no Volume Number. A phony program! See "Making Changes Without DOS BOSS" later on regarding removing the carriage return and blank line after the heading.

4. Quit the Volume Number, and make the heading >>>>> (blank!). Who needs it, anyhow?

### (F) FILE CODE CHANGES

Your A, I, B and T file codes as well as your # and \* lock & unlock codes may be any ASCII screen character you want, Inverse, Flashing, or Normal. Change them as indicated on the screen in the File Code Change mode.

### SUGGESTIONS:

1. Alter your file codes or make them ALL INVISIBLE (spaces) to disguise your files from unauthorized users.

2. Make AppleSoft's A, a J, Integer's I, a >, Binary's B, a #, and Text File's T, a \*. The Key-Cat Program on the DOS BOSS disk uses this set. Make up your own. See Note 4 below.

3. Make the LOCK and UNLOCK codes Inverse to give your catalogs a left border. This helps visually separate columns if you are set up with DOS BOSS's multi-columned catalog.

4. If you are using Key-Cat (page 11), YOU MUST USE ALL STANDARD FILE CODES AND LOCK & UNLOCK CODES.



## (5) SAVING DOS CHANGES

Press 5 from the menu for DOS BOSS's two Save Options.

### QUIT AND INIT

The easiest way to save DOS changes is to create the set-up you like with DOS BOSS, load or create a new "Hello" program, and INIT A NEW DISK with your DOS changes in memory. Booting that disk will format your new DOS for you.

### CREATING POKE FILES

Another way to save your new DOS is to use DOS BOSS to create "Poke-files" which may be appended to any BASIC program. When executed, these pokes will change the values necessary to structure your new DOS.

- First, be sure you have a disk in your drive. Select 5 from the MENU. Then...
  - Press P. You will be asked which TYPES OF CHANGES you want to save: Commands, Error Messages, and/or Catalog changes. The Catalog changes include File Code, Disk Volume Heading, and Column changes.
  - Press Y for each Change Type you want to save, and press S. A text file will be created for each. This will take 30 seconds or so, and you can watch the action.
  - Press Q to exit DOS BOSS and catalog your disk. You will see three text files in the catalog: ERR-POKES, CAT-POKES and COM-POKES.
  - Type NEW, and LOAD or create any BASIC program. BE SURE LINES 30000-30999 ARE NOT USED. That is where data from the Poke Files will be appended.
  - EXEC the appropriate files. EXEC COM-POKES if you are saving Command changes. EXEC ERR-POKES if you are saving Error Message changes. EXEC CAT-POKES if you are saving Catalog Format changes. Each file you EXEC will be appended to your program.
  - Put a GOSUB 30000 in your program and an END before the pokes if necessary. Now this program will format DOS as you have designed it! It can be a "Hello" program if you want. Just delete an existing Hello Program and SAVE this new program under the same name.

Try the above procedure and save your results. There is a program on the DOS BOSS Disk called NORMALIZER. Running it will normalize DOS BOSS changes for you (if you don't want to re-boot).

NOTE: Only NON-STANDARD error messages will be poked in by DOS BOSS's SAVE Feature. All other error messages, including any existing non-standard messages, will remain unchanged.

## Also on the DOS BOSS Disk

### KEY-CAT and BAIT-CAT

STOP!! If you have altered certain DOS Commands in memory with DOS BOSS, rename the strings in LINE 10 of Bait-Cat and Key-Cat or THESE PROGRAMS MIGHT NOT WORK! Your new RUN, BRUN and EXEC commands must be six characters or shorter for Key-Cat. Multi-columns, non-standard file codes and other DOS BOSS changes will also disable Key-Cat.

### KEY-CAT

KEY-CAT is a little BASIC program that will make programs easier to select from the catalog. It is especially handy for non-typers and people not familiar with computers. RUN it and you will see a catalog of your disk. To the left of each file name will be a letter. Press the letter next to the file you want, and Key Cat will RUN, BRUN or EXEC it for you!

If you have more than 17 file names on your disk, the catalog will pause as usual when the screen is full. Press any key to continue. DR, if you see the file name you want on the screen, PRESS THE "RETURN" KEY, and the selection letters will appear.

Type # as your selection and sectors used (SEC USE) and sectors free (SEC FRE) will appear on the screen. Z will let you escape Key-Cat.

Key Cat makes long file names more practical, since you don't have to type the file name to use it. You can use a file name like STAR TREK #9 WITH NEW MONSTERS. The dots on the right of the Key Cat catalog indicate Apple's maximum 30-character file name length.

RENAME KEY-CAT, N and you will only have to type RUNN to change programs. Or RENAME KEY-CAT, DOWN and use RUNDOWN! (Also see NU below).

### BAIT CAT

RUN BAIT CAT to see your files separated by file types: B,A,I and T. If you want to rearrange the order of your files or eliminate certain types, CHANGE CODE IN LINE 150 OF BAIT CAT. Only B,A,I and T are allowed, any order you want.

### LP

LP is a jokey little program to demonstrate a possible use of DOS BOSS's Command Change feature. With DOS BOSS, change BRUN to HE. Then typing HELP will BRUN LP (LP runs a program called ASSISTANCE). You could provide user instructions of your own design this way.

### NU

NU will RUN KEY CAT for you. Change EXEC to ME and the EXEC string in Key Cat (Line 50) to "ME". Now, simply typing MENU will EXEC NU which RUNS KEY CAT!



## Changing DOS Without DOS BOSS

Your Apple's changeable memory (RAM) consists of approximately 48,000 changeable memory locations, (32,000 if 32K). Each location or address is assigned a value from 0 to 255. It is easy to PEEK or look at a value at any location (Example: PRINT PEEK(300) will produce a number, 0-255), and POKE in a new one if you want (Example: POKE 300,123 will change that number to 123). DOS BOSS re-arranges DOS according to your commands by poking or inserting new values into memory for you. Additional possibilities are endless-- Here are a few catalog customizations you can make on your own. Any of these may be entered directly or made part of a program. If you want to keep one of these features, INIT with it poked in, or add the pokes to THE END OF your appended poke-file (see "Saving DOS Changes").

### BEWARE!

Messing around in DOS can cause S-T-R-N-G things to happen, and before you finish this section, you (or I) may have a malfunctioning computer. Fear not! To fix things, simply turn off the power and re-boot. Remember, no permanent harm can ever be done to your Apple itself by just pressing keys.

BEWARE AGAIN! There is also a slight chance that you COULD foul up a DISK as well (if you typed other than the pokes given here), so use an expendable disk (a copy) to play around with. If the disk does get messed up, you can always erase and re-use it by INITing it. Before trying each new poke below, I advise you to poke back in the original values listed after each example.

ATTENTION OUT THERE! The size of your Apple's memory determines where DOS resides. On the following DOS experiments, use the first set of pokes if you have 48K (doesn't everybody?). Use the second set if you have 32K.

-----

### **PRINT PEEK(44611)**

or PRINT PEEK(28227) if 32K (if 32K)

Your Apple should answer with a 2. Let's change it to a 1--

### **POKE 44611,1**

or POKE 28227,1 (if 32K)

Now PEEK again, and there's your 1! Now CATALOG a disk and notice the change you have made. This poke changes the number of digits in your catalog sector numbers to 2 instead of 3. Have you ever seen a file bigger than 99 sectors? Not very often, right? So why clutter up the catalog with extra 0's? (Note: If you REALLY like clutter, poke in a 4 or an 11!) A possible drawback to 2-digit sector numbers: You will scramble your VOLUME NUMBER (on the screen, not in memory). A solution: OMIT the number with DOS BOSS. Another drawback: If you are using DOS BOSS's multi-column catalog, this poke will really make a

temporary mess of your columns! (Normal values: 2. Poke a 2 back in to 44611 or 28227, and continue.)

**POKE 44459, 234; POKE 44460, 234; POKE 44461, 234**  
**POKE 28075, 234; POKE 28076, 234; POKE 28077, 234 (if 32K)**

These two pokes will eliminate the blank line after the word CATALOG. 234's mean "do nothing", and these pokes do nothing instead of printing a carriage return. (Normal values: 32,47,174 if 48K, or 32,47,110 if 32K)

**POKE 44486, 234; POKE 44487, 234; POKE 44488, 234**  
**POKE 28102, 234; POKE 28103, 234; POKE 28104, 234 (if 32K)**

This will eliminate the blank line after the Disk Volume heading. (Normal values: 32,47,174 if 48K, or 32,47,110 if 32K)

**POKE 44452, 24; POKE 44605, 23**  
**POKE 28068, 24; POKE 28221, 23 (if 32K)**

These pokes will let 20 file names appear before stopping for a keypress instead of the normal 18. Poke in any numbers. Always make the first poke value one number larger than the second. (Normal values: 22,21)

**POKE 44541, 173; POKE 44559, 186**  
**POKE 28157, 173; POKE 28175, 186 (if 32K)**

The first poke replaces the space (value 160) after the file-type code with a hyphen. The second one puts a colon after the sector numbers. Experiment with other values from the ASCII Screen Chart in the Dos Boss Book Appendix. (Normal values: 160,160)

**POKE 44567, 12**  
**POKE 28183, 12 (if 32K)**

Will shorten your maximum file name length to 13 characters (on the screen, NOT in memory). Normally this number is 29, for 30 maximum characters. The number of characters is always one less than the number poked in. --y file name shorter than the maximum fills the remaining space with space. (Normal values: 29)

**POKE 44578, 234; POKE 44579, 234; POKE 44580, 234**  
**POKE 28194, 234; POKE 28195, 234; POKE 28196, 234 (if 32K)**

Cancel all carriage returns after file names. With these 234's poked in, play with poking some small numbers (like 2 or 12) only certain numbers will work) in at 44567 (28183 if 32K), and you can have your own version of the multi-column catalog. (Normal values: 32,47,174 if 48K, or 32,47,110 if 32K)



**POKE 44505,234: POKE 44506,234**  
**POKE 28121,234: POKE 28122,234 (if 32K)**  
Shows deleted files in your catalog, and throws in a free bonus inverse character to the right of each. (Normal values: 48,74)

THE FOLLOWING CHANGES ARE PRETTY MUCH WORTHLESS, BUT FUN ANYWAY--

**POKE 44596,234: POKE 44597,234: POKE 44598,234**  
**POKE 28212,234: POKE 28213,234: POKE 28214,234 (if 32K)**  
Will prevent your catalog from stopping when the screen is full. (Normal values: 206,157,179 if 48K, or 206,157,115 if 32K)

**POKE 44599,234: POKE 44600,234**  
**POKE 28215,234: POKE 28216,234 (if 32K)**  
Stops your catalog at each file name, waiting for a keypress on each one (Normal values: 208,8)

**POKE 50,128**  
Makes your catalog invisible (in case you're embarrassed by it). Lists are invisible too! (Normal value: 255)

**POKE 44617,234: POKE 44618,234: POKE 44619,234**  
**POKE 28233,234: POKE 28234,234: POKE 28235,234 (if 32K)**  
Makes all sector numbers and your volume number appear as 000's! (Normal values: 217,164,179 if 48K, or 217,164,115 if 32K)

**POKE 45620,234: POKE 45621,234**  
**POKE 29236,234: POKE 29237,234 (if 32K)**  
Repeats your first file name forever! (Normal values: 105,35)

**FOR X=43439 TO 43443: POKE X,7: NEXT X**  
**FOR X=27055 TO 27059: POKE X,7: NEXT X (if 32K)**

This obnoxious change replaces the word FILE (+ space) with five ctrl-B beeps in the FILE NOT FOUND error message. You can poke beeps (7), carriage returns (13), line feeds (10), or anything into any error message as well as the Volume Message and other unusual places. Each poke, of course, replaces whatever character was there. (Normal values: 70,73,76,69,32)

**POKE 43378,42**  
**POKE 26994,42 (if 32K)**

Now you've replaced all DOS error message beeps (invisible, but audible ctrl-G's) with visible, but inaudible asterisks! (Normal value: 7)

## DOS & Non-DOS Tips and Tricks



**Beagle Bros**  
MICRO SOFTWARE

This section is for fun; a collection of unconnected Apple goodies from here and there, some useful, some useless, ALL kind of interesting.

### IF CITY:

AppleSoft sometimes limits you in your use of "IF" statements. That is, if an IF statement is not true, AppleSoft jumps to the next line number. NOT TRUE if you're dealing with certain types of information! Instead of...

```
10 INPUT A
20 IF A=0 THEN POKE 50,63: REM INVERSE
30 IF A>0 THEN POKE 50,255: REM NORMAL
40 PRINT " BEAGLE "
```

you could say...

```
10 INPUT A: POKE 50,63 + 192 * (A > 0): PRINT " BEAGLE "
```

The IF statement here is really inside the parentheses which takes on a value of 1 or 0 depending on the truth of A>0. IF A>0 then the POKE becomes POKE 50,(63+192) or POKE 50,255 (normal). If not true, it's POKE 50,63 (inverse), AND THE REST OF THE LINE IS STILL READ! A very simple example, but some complex things can be accomplished (like speeding up programs and SAVING SPACE) using this trick.

### FLUSH RIGHT:

Using the method above, flush right numbers are easy. Watch:

```
10 FOR X= 5 TO 1055 STEP 50
20 PRINT SPC((1000): SPC(X(100)): SPC(X(10)))X
30 NEXT X
```

### APPLE BUG DEPARTMENT

(We don't explain 'em; we just find 'em.)

### GET BUG

Turn off DOS by booting with no disk and hitting reset, and type:

```
10 GET G$: V=VAL(G$): PRINT V
```

RUN and enter a digit, 1-9 for G\$. Look at the answer! Inserting a "G\$=G\$" after "GET G\$" seems to clear things up. The explanation for this is really boring. Re-boot to continue...

## SQUARE BUG

PRINT 7 & 7 and PRINT 7 ^ 2 will produce different answers!! Watch this...

```
10 TEXT: HOME
20 PRINT "NUMBER SQUARED" CUSED*
30 PRINT "-----"
40 POKE 34,2
50 FOR I=0 TO 255: PRINT I:
60 HTAB 9: PRINT X^2:
70 HTAB 21: PRINT X^3: NEXT I
```

## CONTROL BUG

Add this line to the beginning of any Applesoft program:

```
10 IF N>99 THEN N=1: PRINT "CATALOG"
```

(There is a control-D hidden between the first quote mark and the C of CATALOG.)

Now try to LIST. You can't! Notice you are presented with a DOS ERROR (SYNTAX ERROR if you haven't used DOS B055) instead of a SYNTAX ERROR. Applesoft can't seem to stand having a ctrl-D at HTAB 1 without trying to EXECUTE the commands following it! To make it execute, REMOVE THE LAST QUOTE MARK (completely legal in Applesoft). Now the LIST command will CATALOG!! The IF N>99, etc., is just filler to get the ctrl-D on the left of the second line. Creative (or destructive) Possibilities: Change CATALOG to FP, and a LIST command will murder the program! Suits that guy right for trying to list your prized SUPERDATABASE program, right? Of course, NO ONE would be ROTTEN enough to change the command to DELETE SUPERDATABASE. (...and INIT SUPERDATABASE? Never!!!)

## ONE MORE NO-LISTER:

A POKE 2049,1 in an Applesoft program will prevent a LIST if the program has been run. Try it in a boot program.

## CALL THIS NUMBER:

CALL-1184 will retrieve a message for you from the Autostart ROM.

## INVISI-CALC?

As mentioned earlier, POKE 50,128 will make a listing or catalog (except for inverse file names) invisible.

This program reveals all the POKE 50 possibilities-- Anyone for secret codes?

```
10 FOR X=0 TO 255
20 POKE 50,255: PRINT "POKE 50,"X:":
30 POKE 50,X: PRINT "TESTING 1,2,3,4,5..."
40 NEXT X: POKE 50,255: END
```

## APPLE'S HEX CONVERTER

You can convert a hex number to decimal in the monitor. Say you want to convert 056E to decimal. From Applesoft, enter the monitor with CALL -151, and type:

```
##5: 05 6E N ED24B (return)
```

Your decimal answer, 1350, is in this case, will appear! Use the N ED24B for converting any number (from Applesoft only).

You can ADD and SUBTRACT hex in the monitor too-- Just type the equation, say "AB+3E" (don't type "PRINT") and hit return, Apple even throws in a free equal sign!

## SHIFTY PICTURES

This little program does tricks with your hi-res screen by shifting memory. First, RUN B.B.L060 from the DOS B055 disk. Enter the monitor with a CALL -151, and type:

```
##2000<2005,JEFFM (return)
```

```
##2000<2010,JEFFM (return)
```

```
##2000<2025,JEFFM (return)
```

These commands tell the Apple to MOVE MEMORY from the address range on the right to the range starting with the address on the left. Substitute your own number after the 'E'. Fun, huh?

## FREEMEM

To find the amount of memory still available to you, PRINT FRE(0) or FRE(Lany number). If you get a negative answer, add 65536. Try printing FRE(0) before and after booting DOS. The program below will show you how many SECTORS a program in memory will occupy according to how much free memory you have left:

```
10 TEXT: HOME
```

```
20 PRINT "SEC MEM" FRE(0):
```

```
30 PRINT "-----"
```

```
40 POKE 34,2
```

```
50 MEM=35329: REM 18945 IF 32K
```

```
60 FOR SEC=0 TO 137
```

```
70 KMEM=1 + INT((.5+SEC)/#255/1000)
```

```
80 MH1=MEM-256#SEC
```

```
90 ML0=MH1-255
```

```
100 PRINT SEC(SEC<100): SPC(SEC<10): SEC: SPC(1)
```

```
110 PRINT SPC(KMEM<10): KMEM: "K": SPC(2):
```

```
120 PRINT ML0: "": MH1:
```

```
130 IF MH1>32767 THEN PRINT " (": ML0-65536: "": MH1-65536: "":
```

```
140 PRINT: NEXT SEC
```



## INVERSE TYPER!

Here's an easy way to type directly to the screen in inverse (or flash):

```
10 INVERSE: REM OR FLASH
20 INPUT A$: PRINT A$: GOTO 20
```

BUT you have to RUN the program and remain in the program for it to work. Run the program below and you will get inverse alphabetical characters after you are out of the program:

```
10 DATA 201,141,240,21
20 DATA 234,234,234,234
30 DATA 201,192,144,13,201,224,176,9,72,132,53,56,233
40 DATA 192: REM 128 FOR FLASH
50 DATA 76,249,233,76,240,253
60 FOR I=768 TO 795: REM $300 TO $318
70 READ N: POKE I,N: NEXT
80 POKE 54,0: POKE 55,3
90 CALL 1002: REM RESET OR PR#0 KILLS THIS PROGRAM.
```

## CONTROL-FIND

Make the following changes to the above program to expose all control characters except U's (forward spaces) and M's (carriage returns):

```
30 DATA 201,128,144,13,201,160,176,9,72,132,53,56,233
40 DATA 128: REM 64=FLASH, 0=NORMAL
```

RUN it (nothing happens). Now type GARBAGE (return) or something misspelled. You will SEE, but NOT HEAR a "SYNTAX ERROR" with an inverse "G" instead of the usual beep! You can type control characters as inverse directly to the screen (or flashing if you change line 40 to DATA 64 in the above program). Backspaces are a bit frustrating to use since they appear as inverse M's and don't visually backspace. Save control characters into your file names if you want, and reveal them by running this program followed by CATALOG.

## INVERSE REM STATEMENTS AT LAST!

After minutes of extensive research, my Uncle Louie finally came up with his finest achievement, INVERSE REM STATEMENTS (flashing if you want!): Maybe not as practical as your normal kind of REM, but they sure do SHOW UP! Here's what you do: Type in the Inverse Typer program from above and enter the CONTROL-FIND corrections. Be sure line 40 is 40 DATA 128. RUN it. Now type an inverse REM in any program. Say, 5000 REM A TEST. Now LIST and you've got it!! Oh, one thing... The characters after "REM" are CONTROL CHARACTERS (ctrl-A, space, ctrl-T, ctrl-E, ctrl-S, ctrl-T). The inverse REMs will only show if you have RUN the control-find program above, hitting reset or PR#0 will kill the effect. Also, your REMs won't be able to contain any M's, U's, or X's (A's & S's too if you use the P.L.E.I.). This is a good way to hide your name or secret info in a program! Thanks, Uncle Louie!

## LEONARDO WOULD HAVE LOVED IT!

You never know when this might come in handy...

```
10 HIS=CHR$(18)+CHR$(18)
20 WTAB 23: HTAB 40
30 GET P$: PRINT P$:H$
40 B010 30
```

## RESET TO THE MONITOR!

We know a guy who chucked his Autostart ROM in the disposal just because it makes RESET return him to BASIC instead of the monitor. He should have typed:

```
3CALL -151
*3F2: 69 FF 5A
Make them 5A a 51 and RESET will RE-BOOT! (Normal is 33F2: 00 97 32)
```

## ODD "T" ~ BELIEVE IT.

File names, according to the DOS Three-Point-Whatever Manual must start with a LETTER. Not so! Shifted letters (M, P, and N) work too, as well as \ and (. Nice for differentiating types of files in your catalogs.

## BUT HOW DO YOU READ THE LABEL?

You will probably ignore this hint! I know I do-- If you are going to leave disks all over your desk, you should place them face UP so that they aren't damaged by dirt, etc. on your desk. The disk drive head reads the BOTTOM of your disk through the oval hole. The hole on top is for the pressure pad.

## AND PEOPLE COMPLAIN ABOUT METRIC!

Have you noticed that to get into this computer stuff, you've got to be constantly CONVERTING things? Decimal to hex, hex to decimal, 3.2 to 3.3, Applesoft to Integer, machine code to BASIC, screen characters to ASCII code, negative memory addresses to positive, 32X to 48k... Good grief!

Here are two rules-of-pinky that I'll pass along at no charge: 4 SECTORS used in a program = approximately 1K of memory (a 24 sector program is about 6K). Also, 4000 decimal = approximately 1000 hex.

## 255 SECTOR HANGSMAN?

You can purposely or accidentally have a file take up more space on a disk than it really occupies in memory. To prove it, SAVE LARGE PROGRAM (say 50 sectors). LOAD TEENY PROGRAM (say 2 sectors). SAVE LARGE PROGRAM. And finally, RENAME LARGE PROGRAM, TEENY PROGRAM. Now TEENY PROGRAM shows 50 sectors in the catalog!

## DELETE HELLO

If you need more space on a disk, consider deleting your Hello Program to save



the amount of space it occupies. You won't be able to boot the disk, but you can still use it:

```
D$ FIX
This program MON'T catalog a disk:
10 D$=CHR$(4): REM CTRL-D
20 PRINT "WATCH...":
30 PRINT D$: "CATALOG"
The problem is that D$ (ctrl-D) in line 30 is at HTAB 9 because of the
semi-colon in line 20. One remedy that always gets your D$'s at HTAB 1 where
it will function is:
10 D$=CHR$(13) + CHR$(4): REM CARRIAGE RETURN + CTRL-D
```

### RIGHT PROTECTORS

Most of the write-protect tabs I have used come off or get messed up going in and out of my disk drives. A handy new product called SCOTCH TAPE works much better! Some computers other than Apple rely on a beam of light to check for write-protection, so their tabs would have to be opaque to work.

### TWO-SIDED DISKS

SURE you can use both sides of your "single-sided" disks. This can save you both money and storage space. Use a regular paper punch to make a half-circle notch on the edge of a disk jacket EXACTLY OPPOSITE the original write-protect notch. Then INIT the second side just like you did the first. Disk manufacturers don't usually test both sides of disks, so you do run the risk of a bad sector now and then and you COULD lose some data. If you make back-up copies anyhow, this shouldn't be a problem. By the way, the small off-center hole in the disk jacket isn't used by Apple's DOS.

### LO-RES MYSTERY

You can't PLOT X,Y where X > 79, BUT you can PRINT SCAN(X,Y) with values of X up to 47! Check it out. It seems that there's an invisible lo-res screen to the right of the visible screen that's 48 plots high by 8 plots wide. Maybe Apple is tooling up for CinemaScope!

### ?=PRINT

AppleSoft thinks ? means PRINT. Try ?2+2. If you use ? in a program and LIST, the ?'s will be converted to PRINTs!

### "QUOTE

AppleSoft doesn't require an end quote mark in most cases. Try PRINT "HELP. (Think of the time you'll save!)

### DE-SCROLLER?

Ever notice that you can't easily print a text character at the lower right of the screen (VTAB:24, HTAB:40)? If you do, everything SCROLLS UP one line. One solution is to POKE in a character. Pick any character from the ASCII Screen Chart in the Appendix (say an Inverse I, code 26) and poke it in during program execution at location 2039 with a POKE 2039,26, NO SCROLL!

### SAVE-PROTECTING YOUR PROGRAMS

There are four basic methods a person will use for copying your software:

1. LOAD and SAVE, after booting your disk.
2. LOAD and SAVE, after booting another disk.
3. Use the F10 program, from the DOS Master Disk.
4. Use a COPY program to copy your entire disk, DOS and all.

Using DOS 5085j, and the following trick, you can foil methods 1, 2 and 3. And method 4 has its drawbacks. The trick involves forcing the user to \$ODT WITH YOUR DOS. The effect is this-- Person P wants to make an unauthorized copy of your fantastic new game. After trying method 1 above, he runs into a (beep) "GAMES NOT COPYABLE" message. He then tries method 2 or 3 which seems to work, but when he tries to run the copy, chaos breaks loose. P is getting discouraged. Perhaps he gives up, or perhaps he goes on to method 4. Method 4 works fine, but (every time P catalogs your disk, he gets a flashing message "(C) JOE JONES!", reminding him of his perfidy. Also, he has to use up a whole disk with your darn personalized DOS on it. P is sorry he ever messed with you!

The procedure to make all this happen goes like this:

- a. Using DOS 5055, replace the READ and SAVE commands. READ becomes SAVE, and SAVE becomes KEEP, or FUI!, or anything you want.
  - b. Replace the NOT INDIRECT COMMAND error message with GAMES NOT COPYABLE.
  - c. POKE 45995,96 (129511,96 if 16K). This will change the sixteenth character of your Disk Volume heading to a flashing space, if you have used the full sixteen characters.
  - d. Personalize your Volume Heading. Make it shorter than 16 characters, so the flashing space doesn't show in the catalog.
  - e. INIT a new disk. Your personal DOS will be on it.
  - f. Copy your programs to the new disk, using copy method 2 or 3 from above.
  - g. Somewhere in each program, insert a CALL 45995 (29611 if 32K). To disguise this call, make it CALL COUNT or something, where COUNT = 45995 or 29611. What this does is insert a CALL to a machine language program that does nothing. But it only does nothing if your DOS is booted with the invisible flashing space installed. For any other DOS, strange and unreasonable things ensue!
- Have fun. And hope that P doesn't have DOS 5055, too!



## FILE NAMES AS TITLES

You have probably noticed our catalog titles in our multi-game disks, where we separate AppleSoft games from Integer:

JCATALOG

DISK VOLUME 123

INTEGER

```
-----
* I 053 TEXTRAIN
* I 036 SUB SEARCH
* I 033 PICK-A-PAIR
```

APPLESOFT

```
-----
* A 053 TEXTRAIN/A
* A 036 SUB SEARCH/A
* A 033 PICK-A-PAIR/A
  etc...
```

The flush left underlined INTEGER & APPLESOFT headings help organize the catalog and separate one group of titles from the other. Here's how we do it:

```
10 DS = CHR$(4); HS = CHR$(8)
20 HS = HS + HS + HS + HS + HS + HS + HS
30 FILES = "Y" + HS + *APPLESOFT*
40 PRINT DS;SAVE *FILES
```

Substitute the word you want for your titles in line 30. You can also replace SAVE in line 40 with DELETE, LOAD or whatever. Access to these "title" is difficult without a program similar to the one above.

Another effective way to make file headings in the catalog is to use inverse or flashing file names. See page 26 of the Winter 80-81 Beagle Bros. Tip Book.

## INPUT ?-REMOVER

In an AppleSoft program, an INPUT A\$ or INPUT A will print a ? and a flashing cursor. If you don't want the ?, change your command to INPUT \*?A\$ or INPUT \*?A.

## SPACE-CAT

Hey everybody! DOUBLE-SPACE YOUR CATALOGS with a POKE 33,37! Mail your reasons for doing this with along with #1 to:

BERT KERSEY  
c/o BEAGLE BROS.  
4315 SIERRA VISTA  
SAN DIEGO CA 92103

## Cruising Through DOS

Let's take a cruise through DOS: The usual "trip" of this kind would be in the monitor, looking at a bunch of two-digit hex numbers; a real drag. Let's make things more interesting! First, boot a normal disk whose DOS hasn't been altered by DOS 80SS. Now write this little program:

```
10 FOR X=43380 TO 43401: REM
   FOR X=26996 TO 27017 IF 32K
20 PRINT PEEK(X); " "
30 NEXT X
```

RUN it, and you will see a string of numbers. Pretty exciting, huh? Now, CHANGE LINE 20 to:

```
20 PRINT CHR$(PEEK(X));
and RUN it again! "LANGUAGE NOT AVAILABLE" magically appears! What's this? You have just uncovered DOS's first error message in the monitor! CHR$(PEEK (X)) means the "character whose ASCII value is X". Now change line 10 to:
```

```
10 FOR X=43380 TO 43581: REM
   FOR X=26996 TO 27197 IF 32K
and you'll see ALL FOURTEEN DOS ERROR MESSAGES strung together! To further examine these mysterious characters, let's add two more lines to our program:
```

```
15 NORMAL: IF PEEK(X)>127 THEN INVERSE
25 IF PEEK(X)>127 THEN PRINT
RUN again, and you'll notice that the LAST CHARACTER of every error message is INVERSE. Actually, line 15 TELLS it to be inverse if it has an ASCII value higher than 127. Each Apple keyboard character and control-character has TWO ASCII values, sometimes called the "low-byte" value and the "high-byte" value. The high-byte character at the end of each error message tells the Apple where the end of the message is and when to quit printing letters to the screen.
```

Now try these immediate mode commands--

```
LOAD ZZZZX
You get a "FILE NOT FOUND", unless you have a program named "ZZZX". Now...
POKE 43452,68
(or POKE 27068,68 if 32K)
```

```
LOAD ZZZZX
You should get a DOUBLE error message, because you have POKED IN or CHANGED the high-byte "g" in "FOUND" to a low-byte "0" (value 68 at location 43452 or 27068). Now the Apple thinks that error message #5 is "FILE NOT FOUNDVOLUME MISMATCH". It prints until it finds a high-byte character, the "H" in "MISMATCH" instead of the "0" in "FOUND"! Repair the damage before continuing
```

## DOS BOSS Listing

by poking the high-byte value for "D" (196) back in where it belongs:  
**POKE 43452,196**  
for **POKE 27066,196** if 32K)

There are other places to look, of course. If you want a really LONG trip through ALL parts of memory, change Line 10 to:

```
10 FOR X=0 TO 65535
```

You will see ALL KINDS of stuff: beeps, backspaces, line feeds, carriage returns, little pictures of animals (just kidding!), and the big feature every so often: REAL WORDS! If you've been running some programs, you'll probably see parts of old program listings. If you've just ERUN or BLOAD'ed the 3.3 MASTER CREATE Program, you'll even find a funny message from the author somewhere between locations 2500 & 3900, only meant to be seen by prying eyes. So go ahead and pry: SAVE your program, BLOAD MASTER CREATE, LOAD your program and RUN.

Back to DOS, change the values in Line 10 to 43140 and 43338 (26756 and 26954 if 32K) and you'll find the 28 DOS commands followed by some garbage and "VOLUME". Those are the initials for VOLUME, DRIVE, SLOT, LENGTH, etc. the one-letter codes used in DOS! Poke in new letter values if you want.

Change Line 10's numbers to 43700 and 43715 (27316 and 27331 if 32K).

That's "APPLESDFT", the name of the program Apple tries to run if you type "FP" and don't have Applesoft in ROM!

Now, change Line 10 to:

```
10 FOR X=46120 TO 45991 STEP-1: REM  
FOR X=29736 TO 29607 STEP-1 IF 32K
```

RUN it and "DISK VOLUME BARRSAIT" appears! That where Apple gets the message to print at the top of your catalogs! "BARRS" seem to be throwaway characters, so we've used them in DOS 8055 to lengthen the "Disk Volume" message. Have you guessed "BAIT" yet? How about Binary, Applesoft, Integer and Text, the codes for your catalog files!

Look around some more! DOS goes clear up to 49151 in 48K; 32767 in 32K. Above 49151 is BASIC. You can look there too if you want! It's YOUR computer!

```
*****  
DOS BOSS  
DISK COMMAND EDITOR  
BY BERT KERSEY  
& JACK CASSIDY
```

```
COPYRIGHT (C) 1981  
BEASLE BROS.  
4315 SIERRA VISTA  
9AM DIEB0 92103  
(714) 296-6400  
*****
```

```
*****  
WARNING!  
DO NOT ALTER ANY LINE >59999!  
PROGRAM WILL BOMB IF YOU DO!  
*****
```

```
110 S0SUB 22500: ONERR G0T0 160  
130 D# = CHR$(13) + CHR$(4):H# = CHR$(8):D# = CHR$(24):HLPH# = "AB  
COEFH1XKLN0PQ6STUWXYZ123"  
140 E(1) = 11E(2) = 23E(3) = 34E(4) = 49E(5) = 60E(6) = 74E(7) = 89E  
(8) = 96E(9) = 107E(10) = 118E(11) = 130E(12) = 150E(13) = 168E  
(14) = 185E(15) = 205  
150 REM
```

```
<M> MENU
```

```
160 TEXT : HOME : INVERSE
```

```
170 VTAB 2: PRINT " "; SPC(12): HTAB 28: PRINT SPC(12): " "; VTAB 22: PRINT  
" "; SPC(38): " "
```

```
180 FOR I = 3 TO 21: VTAB I: HTAB I: PRINT " "; HTAB 40: PRINT " "; I: NEXT
```

```
190 VTAB I: HTAB 15: PRINT " "; " "; HTAB 15: PRINT " DOS BOSS "  
HTAB 15: PRINT " ; " ; NORMAL : PRINT : HTAB 5: PRINT CHR$(IF
```

```
FLAG THEN 250
```

```
200 REM SET-UP COMMAND & MESSAGE STRINGS
```

```
210 VTAB 5: HTAB 16: FLASH : PRINT "C"; NORMAL
```

```
220 FOR I = 1 TO 26: READ X:R# = RIGHT$(X$,1):L# = LEFT$(X$, LEN(X$  
I) - 1):C#(I) = A# + CHR$(ASC(R#) + 128): NEXT
```

```
230 FOR I = 1 TO 14: READ X:R# = RIGHT$(X$,1):L# = LEFT$(X$, LEN(X$  
I) - 1):E#(I) = A# + CHR$(ASC(R#) + 128): NEXT : FLAG = I
```

```
240 VTAB 5: HTAB 16: PRINT "C"
```

```
250 POKE 32,7: PRINT : PRINT " "; INVERSE : PRINT " MENU " ; NORMAL : PRINT
```

```
260 PRINT "(C) DOS COMMAND CHANGE"; PRINT "(E) DOS ERROR MESSAGE CHANGE";
```

```
PRINT : PRINT "(X) CTRL/06 FORMATTER"; PRINT "(V) VOLUME HEADING CHA  
NGE"; PRINT "(F) FILE CODE CHANGE"; PRINT : PRINT "(S) SAVE EXISTING
```

```
SET-UP"; PRINT "(Q) QUIT"
```

```
270 VTAB 19: HTAB 5: S0SUB 5550: GET F#: PRINT F#;" "; POKE 32,0
```

```
280 IF F# = "C" THEN 1000
```

```
290 IF F# = "E" THEN 2000
```

```
300 IF F# = "V" THEN 3000
```

```
310 IF F# = "F" THEN 4000
```

```
320 IF F# = "S" THEN 5000
```

```
330 IF F# = "X" THEN 6000
```

```
340 IF F# = "Q" THEN CALL DING: HTAB 1: PRINT SPC(15): VTAB 23: CALL D
```

```
ING: POKE 216,0: END
```

```
350 CALL DING: G0T0 160
```

```
1000 REM
```

```
<C> DOS COMMAND CHANGE
```

```
3010 TEXT : HOME : INVERSE : HTAB 10: PRINT " "; DOS COMMAND CHANGE : " ; PRINT  
: HTAB 13: PRINT "COMMAND"; HTAB 34: PRINT "COMMAND"; VTAB 3: HTAB
```



```

1100 NORMAL: PRINT "FUNCTION"; HTAB 22; PRINT "FUNCTION"; PRINT "-----"
1020 FOR C = 1 TO 26: GOSUB 1480: HTAB H - 11: PRINT " "
1030 FOR I = 1 TO 132: PEAK(I) = PEEK (MEM + I); CH$(C) = CHR$(C) + CHR$( I
    PEAK(I)); IF PEAK(I) < 128 THEN 1070
1040 X = C: GOSUB 1480: C = C + 1: IF CH$(X) < > CH$(X) THEN HTAB H + 3 -
    LEN (CH$(X)); PRINT " "
1050 PRINT "SAME"
1060 IF X = 26 THEN LTRSUM = 11: = 132: NEXT: GOTO 1080
1070 NEXT
1080 SAME = 0: FOR I = 9 TO 35 STEP 2: SAME = SAME + ( SCREEN( 18, I) > 3) + (
    SCREEN( 39, I) > 3): NEXT
1090 CC = 0: VTAB 19: HTAB 1: CALL - 958: NORMAL: PRINT "-----"
    PRINT "($) STANDARDIZE ALL STANDARD
    SPC( 128 - SAME) < 101: SAME < (N) MENU: HTAB 26: PRINT "CHANGED....."
1100 HTAB 26: PRINT "SPARE CHR$...": IF 132 - LTRSUM > 100 THEN PRINT H
    9
1110 PRINT SPC( 132 - LTRSUM) < 101: 132 - LTRSUM:
1120 FOR C = 1 TO 26: GOSUB 1480: HTAB H - 15: PRINT "($) MID$ (ALPH$, C, 1
    )"; NEXT
1130 VTAB 24: HTAB 5: GOSUB 5550: POKE 34, 23: GET C: PRINT C"; POKE
    34, 0
1140 IF C = "M" THEN 140
1150 IF C = "M" THEN 1450
1160 FOR I = 1 TO 28: IF C = MID$ (ALPH$, I, 1) THEN C = 1: GOTO 1180
1170 NEXT: CALL DING: GOTO 1080
1180 VTAB 20: HTAB 1: CALL - 959: PRINT: INVERSE: HTAB 5: PRINT "STAN
    D BY "
1190 RESTORE: FOR I = 1 TO C: READ C: NEXT
1200 A# = LEFT$ (C$, LEN (C#) - 1): B# = CHR$ ( ASC ( RIGHT$ (C#, 1) ) + 12
    B): CH$(C) = A# + B#
1210 COMMON = C: FOR I = 5 TO 18: VTAB 1: HTAB 1: PRINT " "
1220 F# = CH$(C): NUMBER = 1: FOR I = 1 TO 132: IF PEAK(I) > 127 THEN NUMB#
    R = NUMBER + 1: IF NUMBER = COMMON THEN FIRSTLTR = I + 1
1230 IF NUMBER = COMMON + 1 THEN LASTLTR = I + 1: GOTO 1250
1240 NEXT
1250 IF COMMON = 1 THEN FIRSTLTR = 1
1260 MAX = (132 - LTRSUM) + (LASTLTR - FIRSTLTR): VTAB 20: IF MAX > 9 THEN
    MAX = 9: NORMAL
1270 CALL - 958: GOTO 1290
1280 VTAB 20: CALL - 958: VTAB 24: HTAB 5: PRINT "MAXIMUM "
1290 VTAB 21: HTAB 1: PRINT "ENTER NEW "
    COMMAND: FOR I = 1 TO MAX: PRINT " "
    RETURN: ONLY, IF SAME: VTAB 21: HTAB 19 + LEN (F#): GOSUB
    5560: INPUT " "
1300 TEXT: IF C# = "M" THEN C# = "M" THEN 160
1310 A# = LEFT$ (C#, LEN (C#) - ( LEN (C#) > 1)): X# = CHR$ ( ASC ( RIGHT$
    (C#, 1) ) + 128): IF LEN (C#) = 1 THEN C# = X#: GOTO 1330
1320 C# = A# + X#
1330 EXCESS = MAX - LEN (C#)
1340 IF LEN (C#) > MAX THEN VTAB 20: INVERSE: CALL DING: GOTO 1280
1350 VTAB 20: HTAB 1: CALL - 958: VTAB 21: PRINT " ": FLASH: PRINT "RE
    PLACING "
1360 FOR I = MEM + FIRSTLTR TO MEM + FIRSTLTR + LEN (C#) - 1: POKE I, ASC
    ( MID$ (C#, I - MEM - FIRSTLTR + 1, 1)): NEXT
1370 J = 132 - EXCESS: I = 0: FOR I = MEM + FIRSTLTR + LEN (C#) TO MEM + J
    : POKE I, PEAK(LASTLTR + 1): X = X + 1: NEXT
1380 NUM = 0: FOR I = 1 TO 132: PEAK(I) = PEEK (MEM + 1): IF PEAK(I) > 127
    THEN NUM = NUM + 1
1390 IF NUM = 28 THEN LTRSUM = 11: = 132: NEXT: GOTO 1410
1400 NEXT
1410 IF LTRSUM < 132 THEN FOR I = MEM + LTRSUM + 1 TO MEM + 132: POKE I,
    171: PEAK(I - MEM) = 171: NEXT
1420 C# = COMMON: GOSUB 1480: VTAB V: HTAB H - 11: PRINT " "
    HTAB H - 11: PRINT C#(COMMON): IF C#(COMMON) < > C# THEN HTAB H
    + 3 - LEN (C#): PRINT " "
1430 IF C#(COMMON) = C# THEN HTAB H: PRINT "SAME"
1440 GOTO 1080
1450 VTAB 20: HTAB 1: CALL - 958: VTAB 21: PRINT " ": FLASH: PRINT "RE
    PLACING "
1460 X = 0: FOR I = 1 TO 28: C#(I) = C#(I): FOR J = 1 TO LEN (C#(I)): X =
    X + 1: C = ASC ( MID$ (C#(I), J, 1)): POKE MEM + 1, C: PEAK(X) = C: NEXT
    I: NEXT
1470 FOR C = 1 TO 26: GOSUB 1480: HTAB H - 11: PRINT " "
    H - 11: PRINT C#(C): HTAB H: PRINT "SAME"
1480 V = 4 + C - 14: (C) > 14: H = 16 + 21: (C) > 14: VTAB V: HTAB H: RETURN
2000 REM
<E> ERROR MESSAGE CHANGE
2010 HOME: INVERSE: HTAB 10: PRINT " "
2020 PRINT " "
2030 PRINT " "
2040 FOR I = 1 TO 14: VTAB 4 + I: HTAB 5: PRINT 5: HTAB 5: PRINT EF$(I)
    I: NEXT
2050 EM# = " "
2060 I = J: J = J + 1: IF J > 15 THEN 2110
2070 VTAB 4 + I: IF EM# < > EF$(I) THEN 5# = EM#: GOSUB 240: HTAB 40 -
    LEN (5#): PRINT " "
2080 HTAB 37: PRINT "SAME"
2090 EM# = " "
2100 NEXT I
2110 SAME = 0: FOR I = 9 TO 35 STEP 2: SAME = SAME + ( SCREEN( 3, I) > 3): NEXT
2120 VTAB 19: HTAB 1: CALL - 958: NORMAL: PRINT " "
    SAME < 101: SAME < (N) MENU: HTAB 26: PRINT "CHANGED....."
    SPC( 14
    )
2130 FOR I = 1 TO 14: VTAB 1 + 4: HTAB 1: PRINT "($) MID$ (ALPH$, I, 1)"
    I: NEXT
2140 NORMAL: VTAB 24: HTAB 1: PRINT " "
    CH# PRINT C#": POKE 34, 0: C# = C#: IF C# = "M" THEN 50
2150 IF C# = "M" THEN 2390
2160 VTAB 20: HTAB 1: CALL - 958: PRINT: INVERSE: HTAB 5: PRINT "STAN
    D BY "
2170 FOR I = 1 TO 14: IF C# = MID$ (ALPH$, I, 1) THEN NUM = 5: GOTO 2190
2180 NEXT: CALL DING: GOTO 2110
2190 FOR I = 1 TO 19: VTAB 1: PRINT " "
2200 VTAB NUM + 4: HTAB 2: INVERSE: PRINT " "
2210 EF# = EF$(NUM)
2220 COMMON = NUM: FIRSTLTR = E(CCOMMON): LASTLTR = E(CCOMMON + 9 - 1
    )
2230 MAX = LEN (EF#): NORMAL: GOTO 2250
2240 VTAB 20: HTAB 1: CALL - 958: VTAB 23: HTAB 5: PRINT "MAXIMUM "
    CHARACTERS "

```



```

2250 VTAB 18 20: HTAB 1: PRINT "MEM"; INVERSE; PRINT EF%; NORMAL; PRINT
    *MESSAGE*: VTAB 21: HTAB 1: CALL - 888; VTAB 22: HTAB 5: FOR I = 1
    TO MAX: PRINT "I"; NEXT
2260 VTAB 18 24: HTAB 5: PRINT "1"; RETURN; OR "ONLY", IF SAME; VTAB 22: HTAB
    4: GOSUB 5550: INPUT "C"; IF C = "M" THEN 160
2270 IF C = " " THEN C = EF%;
2280 IF LE LEN (C) < MAX THEN FOR I = LEN (C) + 1 TO MAX: C = C + " ";
    NEXT I
2290 A$ = LEFT$(C$, LEN (C)) - (LEN (C) > 1); B$ = RIGHT$(C$, 1); IF
    ASC (B) (18) < 127 THEN X$ = CHR$(ASC (B) + 128)
2300 IF LE LEN (C) = 1 THEN C$ = X$: GOTO 2330
2310 C$ = A$ + B$
2320 IF LE LEN (C) > MAX THEN VTAB 20: INVERSE; CALL DING: GOTO 2240
2330 VTAB 18 20: HTAB 1: CALL - 958; VTAB 21: PRINT " "; FLASH; PRINT
    *REPLACING*: NORMAL; PRINT " STAND BY,";
2340 FOR I = 1 TO ERLEN + FIRSTLTR TO ERLEN + LASTLTR: POKE I, ASC ( MID$(C
    $, I - ERLEN - FIRSTLTR + 1, 1)); NEXT
2350 VTAB 18 20: HTAB 5: CALL - 888; PRINT 88; HTAB 5: PRINT EF%;
    IF EF% ( > ) C$ THEN S$ = C$: GOSUB 2410: HTAB 40 - LEN (S$); NORMAL
    + PRINT " " + INVERSE; PRINT S$: NORMAL
2360 IF EF% = C$ THEN VTAB 18 20: HTAB 5: PRINT "GAME"
2370 GOTO 2110
2380 VTAB 18 20: HTAB 1: CALL - 958; VTAB 21: PRINT " "; FLASH; PRINT
    *REPLACING*: NORMAL; PRINT " STAND BY,"
2390 X = 0: FOR I = 1 TO 14: FOR J = 1 TO LEN (EF(1)); X = X + 1: POKE ER
    RLEN + I + X, ASC ( MID$(EF(1), J, 1)); NEXT; NEXT
2400 FOR I = 1 TO 14: VTAB 4 + I: HTAB 5: PRINT 88; HTAB 5: PRINT EF(1)
    + HTAB 37: PRINT "SAME"; NEXT; GOTO 2110
2410 FOR I = 1 TO LEN (S$) TO 1 STEP - 1: ASC(1) = ASC ( MID$(S$, I, 1)); IF A
    SKILL = 160 OR ASK(1) = 32 THEN 2430
2420 S$ = LEFT$(S$, I); GOTO 2440
2430 NEXT I: RETURN
2440 RETURN
2500 REM *
<V> VOLUME RE HEADING CHANGE
-----
3010 HOME; INVERSE; VTAB 3: HTAB 10: PRINT "VOLUME HEADING CHANGE";
    NORMAL
3020 VTAB 18 6: HTAB 5: PRINT "EXISTING: "; CALL - 958
3030 VTAB 18 6: HTAB 15: FOR I = MEN + 2871 TO MEN + 2871 - PEEK (MEN + 132
    4) STEP - 1: J = PEEK (I)
3040 IF J = 138 OR J = 136 THEN PRINT CHR$(J); GOTO 3090
3050 IF J = 141 THEN PEEK 32, 14: PRINT CHR$(J); POKE 32, 0: GOTO 3090
3060 IF J > 63 AND J < 128 THEN FLASH; IF J > 95 THEN J = J - 64: GOTO
    3080
3070 IF J < 64 THEN INVERSE; IF J < 32 THEN J = J + 64
3080 PRINT CHR$(J); NORMAL
3090 NEXT I: CALL - 888; IF PEEK (MEN + 134) = 32 THEN PRINT PEEK (ME
    N + 39, 3955);
3100 POKE 32, 0: PRINT; HTAB 15: PRINT "-----"; CALL - 958; PRINT
3110 POZ = PEEK (37) + 1: IF POZ > 20 THEN POZ = 20
3120 VTAB 18 20: HTAB 1: IF PEEK (MEN + 134) = 32 THEN PRINT "<#> OMIT V
    OLUME NUMBER (<#> MENU)"
3130 IF PEEK (MEN + 134) < 32 THEN PRINT "<#> ADD VOLUME NUMBER
    (<#> MENU)"
3140 PRINT "<C> CHANGE HEADING"; PRINT
3150 HTAB 18 5: GOSUB 5550: GET C$: PRINT C$: IF C$ < > * AND C$ < >
    *C* AND C$ < > *M* THEN VTAB POZ: CALL DING: GOTO 3120
3160 IF C$ = "M" THEN 160
3170 IF C$ = "C" AND PEEK (MEN + 134) = 32 THEN POKE MEN + 134, 234: POKE
    MEN + 134, 234: POKE MEN + 134, 234: GOTO 3030

```

```

3180 IF C$ = * AND PEEK (MEN + 134) < 32 THEN POKE MEN + 134, 32:
    POKE MEN + 134, 66: POKE MEN + 134, 174: GOTO 3030
3190 POKE 32, 0: PRINT
3200 VTAB POZ: HTAB 1: CALL - 958
3210 PRINT "NEW HEADING";
    IF PEEK (MEN + 134) = 32 THEN
        VTAB POZ + 2: HTAB 1: PRINT "q = CARRIAGE RETURN " = LINE FEED
        < = BACK SPACE
        > = LEADING SPACE
        N = ME
        NU (CANCEL)
3230 VTAB POZ: HTAB 13: GOSUB 5540: INPUT "C"; IF C$ = "C" THEN POZ: HTAB 15: IF
    LEN (C) > 16 THEN C$ = LEFT$(C$, 16)
3240 PRINT C$: IF PEEK (MEN + 134) = 32 THEN PRINT PEEK (MEN + 3955);
3250 CALL - 888; PRINT; CALL - 958
3260 IF C$ = "M" THEN 160
3270 IF C$ = " " THEN C$ = "DISK VOLUME " + "N"; GOTO 3320
3280 IF LE LEN (C) > 16 THEN C$ = LEFT$(C$, 16)
3290 PRINT "POZ = PEEK (37) + 2
    VTAB POZ - 1: HTAB 1: PRINT "<N> NORMAL (<1>) INVERSE (<F> FLASHING"
    + PRINT; HTAB 28: PRINT "<M> MENU"; HTAB 55: GOSUB 5550: GET A$: PRINT
    A$: IF CALL - 958
3300 IF A$ < > * AND A$ < > *M* AND A$ < > *F* AND A$ < > *N* THEN
    CALL DING: GOTO 3295
3310 IF A$ = "M" THEN 160
3320 POKE MEN + 1324, LEN (C) - 1: POKE MEN + 1328, 187 - LEN (C); FOR
    I = 1 TO LEN (C): S$ = MID$(C$, I, 1): J = ASC (S$) + 128: X = 0
3330 IF X = 0 THEN J = 138: X = 0: GOTO 3420
3340 IF X = 9 THEN J = 141: X = 0: GOTO 3420
3350 IF X = 1 THEN J = 136: X = 0: GOTO 3420
3360 IF X = 0 THEN X = 160
3370 IF A$ = "1" AND (J > 192 AND J < 224) THEN X = 192
3380 IF A$ = "1" AND (J > 192 AND J < 192) THEN X = 128
3390 IF A$ = "F" AND (J > 191 AND J < 224) THEN X = 128
3400 IF A$ = "F" AND (J > 199 AND J < 192) THEN X = 64
3410 IF A$ = "M" THEN X = 0
3420 POKE MEN + 2872 - 1, J - X
3430 NEXT
3440 POZ = PEEK (37) + 3
3450 GOTO 3010
4000 REM
<F> FILE CODE CHANGE
-----
4010 HOME; POKE 32, 8: POKE 33, 32: VTAB 3: HTAB 1: INVERSE; PRINT "FIL
    E CODE CHANGE"; NORMAL
4020 VTAB 6: HTAB 17: PRINT "CODE"; HTAB 17: PRINT "NOW"; HTAB 17: PRINT
    "-----"
4030 I = 2854: PRINT "<A> APPLESOFT, "; GOSUB 4100
4040 I = I - 1: PRINT "<I> INTEGER, "; GOSUB 4100
4050 I = I + 2: PRINT "<B> BINARY, "; GOSUB 4100
4060 I = I - 3: PRINT "<T> TEXT, "; GOSUB 4100
4070 PRINT "PRINT (<L> LOCKED, "; GOSUB 4100
4080 PRINT "(<U> UNLOCKED, "; GOSUB 4100
4090 PRINT; GOTO 4140
4100 J = PEEK (MEN + 1)
4110 IF J > 63 AND J < 128 THEN FLASH; IF J > 95 THEN J = J - 64: GOTO
    4130
4120 IF J < 64 THEN INVERSE; J = J + 64: IF J > 95 THEN J = J - 64
4130 PRINT CHR$(J); NORMAL; RETURN
4140 POKE 32, 8: VTAB 17: HTAB 1: CALL - 958; PRINT; PRINT "<M> MENU"; PRINT
    + PRINT; HTAB 5: GOSUB 5550: GET F$: PRINT F$: IF F$ < > * AND F$ < > *
    * AND F$ < > *U* AND F$ < > *L* AND F$ < > *B* AND F$ < > *I* AND F$ < > *T* AND F$ < > *N* THEN CALL DING: GOTO 4140

```



```

4160 IF F4 = "M" THEN 160
4170 IF F4 = "9" THEN J = 2855:AH = "DINNER":POZ = 11
4180 IF F4 = "A" THEN J = 2854:AH = "APPLESOFT":POZ = 9
4190 IF F4 = "I" THEN J = 2853:AH = "INTEGER":POZ = 10
4200 IF F4 = "1" THEN J = 2852:AH = "TEXT":POZ = 12
4210 IF F4 = "U" THEN J = 1369:AH = "UNLOCKED":POZ = 15
4220 IF F4 = "L" THEN J = 1375:AH = "LOCKED":POZ = 14
4225 VTAB POZ: HTAB 20: INVERSE : PRINT "(-) NORMAL
4230 FOR I = 4 TO 17: VTAB I: HTAB I: PRINT "(-) NEXT I: VTAB 18: HTAB I
: CALL - 958: PRINT " NEW CODE CHARACTER: PRINT " FOR "AH":
"11 HTAB 10 + LEN(AH): PRINT "(-)H#H#H#1 GOSUB 5560: GET C$: PRINT
C$": CALL - 868
4240 IF C4 < " " OR C4 > " " THEN 4230
4245 VTAB POZ: HTAB 18: PRINT C$
4250 TEXT : VTAB 17: HTAB 1: CALL - 958: PRINT "(-) NORMAL (-) INVERSE
(-) FLASHING": PRINT : HTAB 26: PRINT "(-) MENU": HTAB 5: GOSUB 555
0: GET AH: PRINT AH": IF AH = "M" THEN 160
4260 IF AH < " " AND AH > " " AND AH < " " AND AH < " " AND AH < " " THEN CALL DING: GOTO
4290
4270 IF AH = "1" AND (C4 < " " AND C4 < " ") THEN X = 192
4280 IF AH = "1" AND (C4 < " " AND C4 < " ") THEN X = 128
4290 IF AH = "F" AND (C4 < " " AND C4 < " ") THEN X = 128
4300 IF AH = "F" AND (C4 < " " AND C4 < " ") THEN X = 64
4310 IF AH = "M" THEN X = 0
4320 POKE MEM + 3: ASC (C4) + 128 - X
4330 TEXT : GOTO 4010
5000 REM
<S> SAVE EXISTING SET-UP
5010 HOME : PRINT "(-) RETURN TO MENU, OR,": PRINT : PRINT " (-) INVERSE
: PRINT " SAVE EXISTING DOS ONE OF TWO WAYS---": NORMAL
5020 PRINT : PRINT "(-) QUIT THIS PROGRAM & INITIALIZE A NEW DISK WITH
ANY HELLO PROGRAM, BOOTING THAT DISK WILL FORMAT DOS FOR YOU."
5030 PRINT : PRINT " OR,": PRINT : PRINT "(-) P CREATE POKE FILES FOR
APPENDING INTO ANY BASIC PROGRAM,": PRINT
5040 PRINT " (-) GOSUB 5550: GET C$: PRINT C$: IF C4 < " " AND
C4 < " " AND C4 < " " THEN CALL DING: GOTO 5000
5050 IF C4 = "P" THEN 5090
5060 IF C4 = "M" THEN 160
5070 IF C4 = "Q" THEN HTAB 1: PRINT " 1. DELETE DOS BOSS WITH "109"N
EM:109": PRINT
5080 PRINT " 2. LOAD OR CREATE NEW "109"HELLO"109: PRINT " PROS
RAM, APPLESOFT OR INTEGER,": PRINT : PRINT " 3. "99"INIT"99": N
EM DISK,": PRINT : PRINT " OR "109"RUN"109" TO CONTINUE DOS B
OSS,": FOR I = 1 TO 13: PRINT " NEXT I: POKE 216,0: END
5090 VTAB 3: HTAB 1: CALL - 958
5100 VTAB 5: PRINT "SAVE ONE FILE FOR EACH TYPE OF DOS CHANGE Y0
U HAVE MADE": PRINT
5110 PRINT "DOS COMMAND CHANGES . . .": PRINT "ERROR MESSAGE CHANGES . . .
: PRINT "CATALOG CHANGES
5120 F1 = 0:F2 = 0:F3 = 0:J = 8: GOSUB 5160: IF C4 = "Y" THEN F1 = 1
5130 X = 9: GOSUB 5160: IF C4 = "Y" THEN F2 = 1
5140 X = 10: GOSUB 5160: IF C4 = "Y" THEN F3 = 1
5150 GOTO 5220
5160 VTAB 1: HTAB 27: INVERSE : PRINT "(-)SAVE?": NORMAL : PRINT " (Y/N)
: GOSUB 5560: GET C$: PRINT C$
5170 IF C4 < " " AND C4 < " " AND C4 < " " AND C4 < " " THEN CALL DING: GOTO
5180
5180 IF C4 = "M" THEN POP : GOTO 160
5190 VTAB 1: HTAB 27: IF C4 = "M" THEN PRINT "OMIT")
5200 IF C4 = "Y" THEN INVERSE : PRINT " SAVE "1: NORMAL
5210 CALL - 868: RETURN

```

```

5220 PRINT : PRINT : PRINT " (-) SAVE THE ABOVE DOS CHANGES."
: PRINT " (-) RE-SPECIFY,": PRINT " (-) MENU,": PRINT : HTAB 1
0: GOSUB 5550: GET C$: PRINT C$: IF C4 = "M" THEN 160
5230 IF C4 < " " AND C4 < " " AND C4 < " " THEN CALL DING: GOTO 5090
5240 PRINT : PRINT "STAND BY": IF F1 + F2 + F3 = 0 THEN 160
5250 FOR I = 1 TO 28: C4(1) = "": NEXT C4(1) = 1: FOR I = 1 TO 13: C4(1
COMMON) = C4(COMMON) + CHR$(PEEK(MEM + I)): IF PEEK(MEM + I) <
128 THEN 5270
5260 COMMON = COMMON + 1: IF COMMON > 28 THEN I = 132: GOTO 5280
5270 PRINT "": NEXT
5280 C4$ = C4(17): D$ = C4(16): CL$ = C4(9): R$ = C4(10): W$ = C4(12): D
P$ = C4(14): M$ = C4(18): CALL - 868
5290 IF F1 THEN AH = "DOS COMMANDS": C4 = "COM-POKES": LD = 1: HI = 132: LINE
= 30010: GOSUB 5330
5300 IF F2 THEN AH = "ERROR MESSAGES": C4 = "ERR-POKES": LD = 24: HI = LD +
20: LINE = 30100: GOSUB 5330
5310 IF F3 THEN AH = "CATALOG FORMAT": C4 = "CAT-POKES": LD = 139: HI = 144
: LINE = 30200: GOSUB 5330
5320 GOTO 160
5330 PRINT D$: M$: " C,1,0": PRINT : INVERSE : PRINT " SAVING "AH": " NORMAL
5340 PRINT D$: D$: "1C$: PRINT D$: D$: "1C$: PRINT D$: D$: "1C$: PRINT
D$: W$: "1C$
5350 PRINT "30000 M:(PEEK(978)-(PEEK(978)>127)&256)&2947: E=M+240": PRINT
"30900 RETURN"
5360 PRINT LINE: " REM "AH: LINE = LINE + 1: NUM = 0
5370 IF C4 = "ERR-POKES" THEN 5480
5380 FOR I = LD TO HI: NUM = NUM + 1: IF NUM = 9 THEN NUM = 1
5390 IF NUM = 1 THEN PRINT : PRINT : PRINT LINE: LINE = LINE + 1
5400 PRINT " POKE M+"11": PEK (MEM + I)1": NEXT I: PRINT
5410 IF C4 < " " AND C4 < " " AND C4 < " " THEN 5460
5420 IF LINE < 30300 THEN LINE = 30300: LD = 2852: HI = 2971: NUM = 0: GOTO
5380
5430 IF LINE < 30400 THEN PRINT "30400 POKE M+"1324": "1: PEK (MEM + 13
24)1": POKE M+"1326": "1: PEK (MEM + 1326)
5440 IF LINE < 30500 THEN LINE = 30500: LD = 1341: HI = 1343: NUM = 0: GOTO
5380
5450 IF LINE < 30600 THEN PRINT "30600 POKE M+"1369": "1: PEK (MEM + 13
69)1": POKE M+"1376": "1: PEK (MEM + 1376)
5460 PRINT D$: CL$: "1C$
5470 RETURN
5480 FOR C = 1 TO 14: E%(C) = "": FOR I = E(C) TO E(C) + 1 - 1: E%(C) = E
%(C) + CHR$(PEEK(ERMEN + I)): NEXT I: NEXT
5490 FOR C = 1 TO 14: IF E%(C) < " " AND E%(C) < " " AND E%(C) < " " AND E%(C) < " "
: GOSUB 5510
5500 NEXT I: GOTO 5460
5510 FOR I = LD TO HI: NUM = NUM + 1: IF NUM = 9 THEN NUM = 1
5520 IF NUM = 1 THEN PRINT : PRINT : PRINT LINE: LINE = LINE + 1
5530 PRINT " POKE E+"11": "1: PEK (ERMEN + I)1": NEXT
5540 RETURN
5550 PRINT "SELECT: < "1": "1
5560 FOR I = 1 TO 2: FOR K = 1 TO 5: IZ = PEEK (- 16336) + PEEK (- 163
36)1: NEXT I: FOR K = 1 TO 50: NEXT I: NEXT I: POKE - 16368,0: RETURN
6000 REM
<X> CATALOG FORMAT
6010 TEXT : HOME : INVERSE : VTAB 3: HTAB 10: PRINT " CATALOG FORMATTER
: NORMAL
6020 FIRSTLR = 0: LASTLR = 0: J = 1: FOR I = 1 TO 132: IF PEEK (MEM + I) <
128 THEN 6050
6030 J = J + 1: IF J = 17 THEN FIRSTLR = I + 1
6040 IF J = 18 THEN LASTLR = 1: GOTO 6060

```







# ASCII Screen Characters

Here is a listing of Apple's 256 ASCII Screen Characters, adapted from page 15 of the Apple II Reference Manual. Each character is followed by its hexadecimal & decimal values and the "to-byte" and "high-byte" decimal values.

## INVERSE

2=40=000 64-192	A=401=001 65-193	B=402=002 66-194	C=403=003 67-195
D=404=004 68-196	E=405=005 69-197	F=406=006 70-198	G=407=007 71-199
H=408=008 72-200	I=409=009 73-201	J=40A=010 74-202	K=40B=011 75-203
L=40C=012 76-204	M=40D=013 77-205	N=40E=014 78-206	O=40F=015 79-207
P=410=016 80-208	Q=411=017 81-209	R=412=018 82-210	S=413=019 83-211
T=414=020 84-212	U=415=021 85-213	V=416=022 86-214	W=417=023 87-215
X=418=024 88-216	Y=419=025 89-217	Z=41A=026 90-218	[=41B=027 91-219
\=41C=028 92-220	]=41D=029 93-221	^=41E=030 94-222	_=41F=031 95-223

=420=032 32-160	!421=033 33-161	"422=034 34-162	#423=035 35-163
\$424=036 36-164	%425=037 37-165	&426=038 38-166	'427=039 39-167
(428=040 40-168	)429=041 41-169	*42A=042 42-170	+42B=043 43-171
,42C=044 44-172	-42D=045 45-173	.42E=046 46-174	/42F=047 47-175
0430=048 48-176	1431=049 49-177	2432=050 50-178	3433=051 51-179
4434=052 52-180	5435=053 53-181	6436=054 54-182	7437=055 55-183
8438=056 56-184	9439=057 57-185	!43A=058 58-186	"43B=059 59-187
<43C=060 60-188	=43D=061 61-189	>43E=062 62-190	?43F=063 63-191

2=440=064 64-192	A=441=065 65-193	B=442=066 66-194	C=443=067 67-195
D=444=068 68-196	E=445=069 69-197	F=446=070 70-198	G=447=071 71-199
H=448=072 72-200	I=449=073 73-201	J=44A=074 74-202	K=44B=075 75-203
L=44C=076 76-204	M=44D=077 77-205	N=44E=078 78-206	O=44F=079 79-207
P=450=080 80-208	Q=451=081 81-209	R=452=082 82-210	S=453=083 83-211
T=454=084 84-212	U=455=085 85-213	V=456=086 86-214	W=457=087 87-215
X=458=088 88-216	Y=459=089 89-217	Z=45A=090 90-218	[45B=091 91-219
\=45C=092 92-220	]=45D=093 93-221	^45E=094 94-222	_45F=095 95-223

=460=096 32-160	!461=097 33-161	"462=098 34-162	#463=099 35-163
\$464=100 36-164	%465=101 37-165	&466=102 38-166	'467=103 39-167
(468=104 40-168	)469=105 41-169	*46A=106 42-170	+46B=107 43-171
,46C=108 44-172	-46D=109 45-173	.46E=110 46-174	/46F=111 47-175
0470=112 48-176	1471=113 49-177	2472=114 50-178	3473=115 51-179
4474=116 52-180	5475=117 53-181	6476=118 54-182	7477=119 55-183
8478=120 56-184	9479=121 57-185	:47A=122 58-186	;47B=123 59-187
<47C=124 60-188	=47D=125 61-189	>47E=126 62-190	?47F=127 63-191

## CONTROL

2=80=128 0-128	A=81=129 1-129	B=82=130 2-130	C=83=131 3-131
D=84=132 4-132	E=85=133 5-133	F=86=134 6-134	G=87=135 7-135
H=88=136 8-136	I=89=137 9-137	J=8A=138 10-138	K=8B=139 11-139
L=8C=140 12-140	M=8D=141 13-141	N=8E=142 14-142	O=8F=143 15-143
P=90=144 16-144	Q=91=145 17-145	R=92=146 18-146	S=93=147 19-147
T=94=148 20-148	U=95=149 21-149	V=96=150 22-150	W=97=151 23-151
X=98=152 24-152	Y=99=153 25-153	Z=9A=154 26-154	[9B=155 27-155
\9C=156 28-156	]=9D=157 29-157	^9E=158 30-158	_9F=159 31-159

## NORMAL

=40=160 32-160	!401=161 33-161	"402=162 34-162	#403=163 35-163
\$404=164 36-164	%405=165 37-165	&406=166 38-166	'407=167 39-167
(408=168 40-168	)409=169 41-169	*40A=170 42-170	+40B=171 43-171
,40C=172 44-172	-40D=173 45-173	.40E=174 46-174	/40F=175 47-175
0408=176 48-176	1409=177 49-177	240A=178 50-178	340B=179 51-179
4404=180 52-180	5405=181 53-181	6406=182 54-182	7407=183 55-183
8408=184 56-184	9409=185 57-185	:40A=186 58-186	;40B=187 59-187
<40C=188 60-188	=40D=189 61-189	>40E=190 62-190	?40F=191 63-191

2=4C0=192 64-192	A=4C1=193 65-193	B=4C2=194 66-194	C=4C3=195 67-195
D=4C4=196 68-196	E=4C5=197 69-197	F=4C6=198 70-198	G=4C7=199 71-199
H=4C8=200 72-200	I=4C9=201 73-201	J=4CA=202 74-202	K=4CB=203 75-203
L=4CC=204 76-204	M=4CD=205 77-205	N=4CE=206 78-206	O=4CF=207 79-207
P=4D0=208 80-208	Q=4D1=209 81-209	R=4D2=210 82-210	S=4D3=211 83-211
T=4D4=212 84-212	U=4D5=213 85-213	V=4D6=214 86-214	W=4D7=215 87-215
X=4D8=216 88-216	Y=4D9=217 89-217	Z=4DA=218 90-218	[4DB=219 91-219
\4DC=220 92-220	]=4DD=221 93-221	^4DE=222 94-222	_4DF=223 95-223

## LOWER CASE

=4E0=224 96-224	a4E1=225 97-225	b4E2=226 98-226	c4E3=227 99-227
d4E4=228 100-228	e4E5=229 101-229	f4E6=230 102-230	g4E7=231 103-231
h4E8=232 104-232	i4E9=233 105-233	j4EA=234 106-234	k4EB=235 107-235
l4EC=236 108-236	m4ED=237 109-237	n4EE=238 110-238	o4EF=239 111-239
p4F0=240 112-240	q4F1=241 113-241	r4F2=242 114-242	s4F3=243 115-243
t4F4=244 116-244	u4F5=245 117-245	v4F6=246 118-246	w4F7=247 119-247
x4F8=248 120-248	y4F9=249 121-249	z4FA=250 122-250	[4FB=251 123-251
\4FC=252 124-252	]=4FD=253 125-253	^4FE=254 126-254	_4FF=255 127-255

File Codes, Disk Volume Headings, and the like, use the first set of numbers. Commands and Error Messages use the High and Low-Byte set. This chart also serves as a handy Hex-Decimal conversion chart for numbers 400 to 9FF (0-255).

# Beagle Bros. \$24 Apple Game Packs!

Now you can buy an assortment of exciting Beagle Bros. Apple games on ONE DISK and save some bucks! **THREE GAMES PLUS\*** are included on each 3.2 or 3.3 disk, most in Applesoft. AND Integer Basic.\*\* for the Total Price of **ONLY \$24!** An 11 x 17 **APPLE COMMAND CHART** and 36-page **APPLE TIP BOOK** are included FREE!



FREE WITH EVERY DISK!



Free 11 x 17 Beagle Bros. COMMAND CHART and 36-page APPLE TIP BOOK included with each disk!

## GAME PACK #1

(Advanced and Integer)



- 1. TEXTTRAIN:** Race the on-screen clock with your text-format video " freight train." Real-time track switching and changing situations and hours of fun!
- 2. SUB SEARCH:** Find the invisible enemy subs on your Apple color graphics scope before your oxygen & fuel run out! Sound-enhanced laser, tracer and animated instrument panel!
- 3. PICK-A-PAIR:** A colorful Apple party game and a proven winner! Uncover and remember the hidden graphics symbol to score big and well!

## GAME PACK #2

(Advanced and Integer)



- 1. WOWZO!** Beagle Bros.' challenging changeable maze game! Search and capture targets in a Beagle maze, and outmaneuver your opponent before time runs out!
- 2. ELEVATORS:** Keyboard control four elevators at one time in your CAT skyscraper. You'll need a controller to solve the one!
- 3. QUICK-DRAW!** Two colorful gunmen shoot it out on your Apple screen!

## GAME PACK #3

(Advanced and Integer)



- 1. MAGIC PACK:** Four brain-bending tricks in one great magic show! Only you and your Apple know how these amazing tricks are done!
- 2. SLEEPY DIGITS:** A challenging and colorful number-action game for all ages. A GREAT demonstration of your Apple's capabilities!
- 3. OINK!** A never-ending video dice game with unpredictable results!

## GAME PACK #4

(Advanced and Integer)



- 1. BUZZWORD:** A favorite with kids of all ages! Your Apple types a story and you choose the missing words. OR you create a story with Apple's help. Save your stories to disk for future laughs!
- 2. TRIPLE DIGITS:** A play-ahead numbers game, easy to play, but a real challenge for anyone! Stimulates some real thinking about numbers.
- 3. CORN GAME:** For kids, a guessing game involving three farm animals and an endless supply of CORN!

\*In addition to the games listed here, each disk contains AT LEAST TWO additional Apple BASIC programs, powerful new graphics and sound routines to maximize the versatility of your Apple.

\*\*Disk BASIC and Integer BASIC are a nice upgrade to Applesoft BASIC. Additional programs are available for purchase separately.

Copyright © 1981, Beagle Bros. Micro Software  
4115 Sierra Vista, San Diego, CA 92103, (714) 936-6400



Mail to: 4315 Sierra Vista / San Diego, CA 92103