```
|----------------------------HELP for HLP-----------------------------|
| use:BATHLP <DOS cmd> brings Help or reports no help is available.   |
| how:Type "BATHLP <DOS cmd>"to get help for commands MARKED "h" BELOW |
|  b AUTO-1.HLP  |  * BATHLP.HLP  |  h IF.HLP       |  h PAUSE.HLP    |
|  b AUTO-2.HLP  |   *this file   |  h IFEXIST.HLP  |  h REM.HLP      |
|  b AUTO-3.HLP  |  h ECHO.HLP    |  b INTRO-1.HLP  |  h SHIFT.HLP    |
|  b BATDO-1.HLP |  h EDLIN.HLP   |  b INTRO-2.HLP  |  h SORT.HLP     |
|  b BATDO-2.HLP |  h FIND.HLP    |  b INTRO-3.HLP  |  h STOPBAT.HLP  |
|  b BATDO-3.HLP |  h FOR.HLP     |  b INTRO-4.HLP  |  h VAR.HLP      |
|  b BATDO-4.HLP |  h FORINDO.HLP |  h MORE.HLP     ||  >THIRTY .HLP  |
|  b BATDO-5.HLP |  h GOTO.HLP    |  h PARAM.HLP    ||  >FILES TOTAL  |
|     LEGEND:  h=.HLP file  !=1/line descr.   b=run by BATCHLRN       |
| N-O-T-E: For a "quickie" SHORTENED listing type: "QWIKBAT"!!  For a |
| printed version (To Print ALL SCREENS) Type:"PRINT BATHLP.PRN"      |
|   WHEN PRINTING >do not< FORGET TO TYPE: PERFSKIP 6  >f-i-r-s-t<    |
|---------------T  I  M  E  M  A  S  T  E  R-----------------|
|---------------B A T C H L R N  H E L P  S Y S T E M-----------------|
command: ECHO
use: The batch command ECHO lets you choose whether or not to have
 the commands in your batch file displayed on the CRT screen as they
 are performed. ECHO is also used for message display (no commands
 are shown). SEE NOTES SECTION BELOW.

how: Type: ECHO ON    O-R ECHO OFF [for message sequencing]

examples: REFER TO ANY TIMEMASTER .BAT FILES ON THIS DISK

notes: Commands in a batch file are normally displayed as they are
 performed by your computer. If you enter ECHO OFF into a batch file
 the commands are not displayed when they are performed. Command dis-
 play while a batch processing file is operating can be distracting.
 If you display the commands have a good reason for it!
     A little known (useless?) feature is to type: ECHO <ret> at the
 prompt, which displays current status of ECHO. If you type ECHO OFF
 <ret> the prompt will disappear. Type:ECHO ON to restore!
     If ECHO is set to OFF,you can use ECHO <your message>to display
 text on screen. You can use this ONLY in a batch processing file.
 ECHO sends a message to the CRT screen. It helps organize and  ex-
 plain on the screen when a batch file executes. Normally, all batch
 file commands would show on the screen as the file performs. To pre-
 vent this echoing type:  ECHO OFF at the beginning. In particularly
 long batch files you may have to do this after several or all of the
 message echoeing sequences. This becomes especially true when you
 are commanding the display of multiple page ascii text files or
 when showing long files a page at a time with the DOS MORE command.
 Your test run of the batch file you create will show you whether or
 not additional ECHO OFF cmds are necessary. We always"follow-on"the
 ECHO OFF with a CLS on the next line. This keeps the on-screen co-
 mmand distraction to a minimum. If you examine the .BAT & .EX files
 on this disk you will see the use of ECHO in TIMEMASTER'S efforts.
     When ECHO is OFF none of the actual DOS commands in the batch
 file will show on the screen. To display a message put it after tbe
 command:          ECHO  . . .MY DOG HAS FLEAS!. . .
 (Watch the placement, trying for message centering on the screen).
     As you will see in REM.HLP,that command also displays messages.
 You will learn,however, that is not the case if ECHO is OFF.

---------------- T  I  M  E  M  A  S  T  E  R --------------------
|---------------B A T C H L R N  H E L P  S Y S T E M-----------------|
command:EDLIN (with many, many features)
```

----------ONE-LINER HLP----------

use:USE OF THIS DOS COMMAND WILL REQUIRE REFERRING TO YOUR MANUAL.*

EDLIN:Line editor to create,change and display files of ALL kinds.

examples:ALL of the batch files on this disk were created using ED-
LIN. Should you desire to make changes you can do it the same way!

*N-O-T-E:Many DOS Manuals scare newcomers with:"Edlin is not inten-
ded for novice users." BALONEY! Experiment with EDLIN until you
know HOW TO USE IT! It can become one of your BEST FRIENDS! To the
BATCH PROCESSING PROGRAMMER (we don't care that THEY say"batching"
is NOT programming) it allows you flexibility and fun! We substan-
tially PREFER EDLIN to COPY CON for creating batch processing files
since most of the .BAT files we create are long AND involved. Con-
making it a rule to use EDLIN if the batch file will be more than
two or three lines.

----------T I M E M A S T E R----------
----------B A T C H L R N H E L P S Y S T E M----------
command:FIND
use:FIND searches a file/DIR for a specified word or text string.

how:Type "FIND <insert option> <word or string> <file.ext>".
o-r DIR|FIND "MY"will list all files in curr.dir w/"MY"(caps ONLY!)

examples:Look at the BATSEEK.BAT file for an example of the FIND
command used in a batch processing file in conjunction with other
commands and functions. Like any file being used for an example,
however, it will require s-t-u-d-y! Print the file out and examine
the hard copy carefully.

N-O-T-E-S:Options [/C=display ONLY lines with string/word.
[/V=display all lines NOT containing word/string].
[/N=display ONLY matches (with line numbers).
You can search several files @ once:FIND "my" file1.ext file2.ext.
will search for "my" in BOTH of those files (ONLY lower case ltrs).
Upper and lower case letters are NOT CONSIDERED A MATCH {If you
want"FOO" DO NOT type"foo"}. Tip:Type "listing" files in ALL CAPS.
TIMEMASTER has utility files which convert ENTIRE FILES, either to
all caps or all lowercase, by the use of simple commands.
----------T I M E M A S T E R----------
----------B A T C H L R N H E L P S Y S T E M----------
command: FOR
use: The FOR command is used to expand the power of a batch file
command or as an interactive (from the keyboard prompt) command
how: In Batch processing type: FOR %%<c>IN<set>DO<command>
In Interactive processing type: FOR %<c>IN<set>DO<command>
(Did you say Huh? You demand...and get, a translation...)
translation: %%<c> or %<c> is a variable. <c> can be any character
except 0,1,2,3...,9. This avoids confusion with the %0-%9 batch pa-
ramaters. <set> is a list of items separated by spaces and enclosed
in parentheses. For example:: (item1 item2 item3...itemN) is a set.
The %%<c> variable is set sequentially to each member of the <set>,
and then <command> is performed. If a member of <set> is an express-
ion involving either * or ?, or both, then the varible is set to
each matching pattern from the file. In this case, only one such
<item> may be in the set. Any <item> other than the first <item> is

ignored.
(We KNOW this is DIFFICULT, but STAY with us!). . .
explanation: The translation is complex,but UNTIL YOU UNDERSTAND IT, SO is the FOR command. For that reason,AFTER you have finished this .HLP file go to FORINDO.HLP which has examples & explanations(They are NO GOOD, however, unless you understand the overall concept)!
The FOR command is similar to the "for..next loop" that programmers use to repeat a paricular action a given number of times. The FOR file to compare files on A: & B: drives looks like this:

```
FOR %X IN (*.*) DO IF EXIST B:%X ECHO %X IS ON BOTH A:&B:
    |       |         |=the<command>(here, starting on A:drive, the
    |       |           DO sees IF X {which is *.*[any of all the
    |       |           files on A:]} exists on B: the ECHOes the
    |       |           file name {X} as being on "BOTH A:&B:")
    |       |
    |       |=IN<set> the set is *.* or all the files on A:&B:
    |
    |=FOR is seeking the variable %X (it could be any letter)
```

(It should be starting to get clearer)
What we have displayed is an interactive FOR command string. If it were in a two-line batch file (for example) the first line would be ECHO OFF to keep command clutter off the screen. ALSO,in a batch file %X would be %%X (this tells DOS this is NOT a marker. It also leaves one % after multiple parameter processing {seeFORINDO>HLP}).
examples: SEE THE FORINDO.HLP FILE FOR EXAMPLES & EXPLANATIONS!

notes: Remember, in a batch file, you must use the expression "%%". If you are in interactive DOS processing mode,only ONE % is needed. You cannot nest FOR commands in DOS like the FOR-NEXT command is used in GWBASIC. If you try to do this, the message:
"FOR cannot be nested"
appears when you are running your batch program. The program will not perform as you expected.
(SET) is one or more filenames or commands you want %%Varible to assume while the command is being executed. Use a space between entries, and pathnames are not allowed. Don't folrget the parenthesis around the set. Wildcards may be used within (SET) if you are using filenames.
Command is the particular DOS command or batch subcommand you want to have performed. Usually one or more of these commands will contain the %%Varible in it. If (SET) contains DOS commands, only %%Varible is used.
In effect, what this subcommand does is cause %%Varible to be an index into the (SET) for use by command.

---------- T I M E M A S T E R ----------
----------B A T C H L R N H E L P S Y S T E M----------
command: FOR..IN..DO (Actually multiple commands)
use:FOR..IN..DO string is used to expand the power of a batch file or as an interactive (from the keyboard prompt) command

how: In Batch processing type: FOR %%<c>IN<set>DO<command>
In Interactive processing type: FOR %<c>IN<set>DO<command>

explanation: The FOR command is similar to the "for..next loop" that programmers use to repeat a paricular action a number of times. A batch file to compare files on A:& B: drives looks like this:

```
ECHO OFF        [keeps command clutter of the screen]
CLS             [we always do this to get ECHO OFF removed]
```

```
    FOR %%Y IN (*.*) DO IF EXIST B:%%Y ECHO %%Y IS ON BOTH A:&B:
     |       |        |=the<command>(here, starting on A:drive,the
     |       |         DO sees if X {which is *.*[any of all the
     |       |         files on A:]} exists on B: the ECHOes the
     |       |         file name {X} as being on "BOTH A:&B:")
     |       |
     |       |=IN<set> the set is *.* or all the files on A:&B:
     |
     |=FOR is seeking the variable %X (it could be any letter)


examples: FOR %%gf IN(*.ASM)DO MASM %%f;
 FOR %%f IN(BAK ART BUDGT) DO REM %%f
 The "%%" is needed so that after batch parameter (%0-%9) processing
 is complete, a % remains. If only %f was entered, the DOS batch pa-
 rameter processor sees the first "%", looks at "f", then decides
 that "%f" is a bad parameter reference, and discards the "%f".
 The FOR command would never receive this parameter. In a batch
 file, you must use the expression "%%".
 Let's see hoe the batch command works... The first line turns
 command displat off to clear command clutter. The second line clears
 the "echo off" messagefrom the screen. The third line is executed
 many times as there are files on the disk in a: [the set (*.*)
 assures this]. Each of those filenamesare assigned to %%Z in turn
 and then checked for presence on drive B: with the EXIST logical
 statement. If EXIST is true, then the message at the end of the IF
 subcommand si sent to the screen, otherwise nothing is printed and
 the next file on driove A: is assigned and checked.
   FILES on drive A:              FILES on drive B:
   -----------------              ------------------
     COMMAND.COM                    COMMAND.COM
     FILE.ONE                       FILE.ONE
     FILE.TWO                       FILE.LTR
 The batch subcommand we are investigating is:
 FOR %%Z IN (*.*) DO IF EXIST B:%%Z is on both A: and B:
 Each filename on A: is substitutedin the IF subcommand and then
 executed. To get the same effect you would have to type--
 IF EXIST B:COMMAND.COM ECHO COMMAND.COM is on both A: and B:
 IF EXIST B:FILE.ONE ECHO FILE.ONE is on both A: and B:
 IF EXIST B:FILE.TWO ECHO FILE.TWO is on both A: and B:
 In the case of the example above, the first two would have a posi-
 tive responseand the last would not print anything. Study it care-
 fully before going on.
           FILES on drive A:             FILES on drive B:
           -----------------             -----------------
            COMMAND.COM                   COMMAND.COM
            FILE.ONE                      FILE.ONE
            FILE.TWO                      FILE.LTR
 OK, told you to study the example. Let's see if you remember. What
 is the one line batch file command to find and report out all files
 starting with an "F" on drive A:? [I'll help you a little, you fill
 in the blanks.]
 ___%%Z IN (A:_____) ____ECHO File_____is on drive A:
 FOR %%Z IN (A:F*.*) DO ECHO File %%Z is on drive A:
 In this case you see that the A: disk is checked for any file start-
 ing with the letter "F" and that name is substituted in the varible
 %%Z. The appropriate message is then printed with the proper file-
 name.
 This is not an easy concept. Let's now look at using DOS commands
 in (Set).
```

FOR...IN..DO using DOS Commands
==========

The set can contain DOS commands instead of filenames and these commands will then be executed in sequence (to include running large programes under the control of the FOR..IN..DO loop). Let's say you want to sequentially: Clear the screen
Show the DOS version number
then Show the disk directory with the pause on
You could do all that in a one line batch file with the following command in it:: FOR %%T IN (CLS VER DIR/P) DO %%T
When using DOS commands in (SET) ypu must use the space as a de-limiter and cannot have asterisk (*) or question mark (?) in any command. Use a colon(:) instead of a space when passing parameters to programs (i.e. DBASE:FILE INSTEAD OF DBASE FILE)
It is possible to issue the FOR..IN..DO command at the DOS prompt by dropping one of the percentage signs (%) an the varible.

---------- T I M E M A S T E R ----------
----------B A T C H L R N H E L P S Y S T E M----------

command: GOTO
use: The GOTO command is a batch file-only command. It is used to perform a "jump" from the GOTO <label> to the :<label>. After the :<label> (the SECOND one) the program performs the commands listed. In"computereze"this is described as:the GOTO command directing the program to perform the command following the item described in the command modifier (so much for that!).

how:Type: GOTO <LABEL> (in a batch file), THEN type:<LABEL> later on for a series of commands to be performed.

when:GOTO is used in conjunction with (usually) some IF condition or to create a loop function.

examples: In a condition situation you would have SEVERAL GOTOs. For example, IF %1 exists GOTO label 1 (next line) IF %1 not exist GOTO label#2. Elsewhere in the batch file :label#1 AND :label#2 would both be present, followed by commands you want performed depending on whether %1 existed or not. USUALLY you create another jump at the end of each :label command set with GOTO END, with the :END label wrapping up the batch processing no matter what else occured in the previous portions of the file. SEE sample batch files on this disk!
more examples:(To create a GOTO loop): 1) :spot
this is what you would 2) REM This is a loop
program (3 lines) 3) GOTO spot
That would cause an infinite sequence of messages to be displayed:
A>REM This is a loop
A>GOTO spot
A>REM This is a loop
A>GOTO spot
A>REM This is a loop . . . . etcetera (on & on til you stop it!)
NOTES: If you do not include <label> in the GOTO command, the batch processing file stops. Any line in a batch file that starts with ":" is treated as a label but otherwise ignored. DOS only recognizes the FIRST EIGHT characters in a label. To stop an endless loop GOTO file use CTRL-BREAK or CTRL C.

---------- T I M E M A S T E R ----------
----------B A T C H L R N H E L P S Y S T E M----------

command: IF
use: The IF subcommand allows you to direct the decision-making pro-cess in batch processing files (referred to as statement "testing").

how: Type: IF <condition> <DOS or batch command> [see*&**NEXT]
 **<command> == Any legal DOS command or batch file subcommand
 *<condition> == One of three tests that yield a true or false re-
 sult:
                1. The ERRORLEVEL of a program (see below)
                2. Two strings that are equivalent or the same.
                3. A file that exists in the CURRENT directory.

explanation:The IF command tests for true or false. IF true, it per-
 forms the command. IF NOT true,it skips the command and goes to the
 NEXT line in the batch processing file. You can "test" SEVERAL lines
 one after the other, so that if the first line is not true the batch
 file moves to the next "testing" line. This will continue until no
 more tests are left or a TRUE statement directs a command(like GOTO)

examples: Refer to BATSEEK.BAT to see the file testing for the pres-
 ence of.EXE &.COM files, the GOTO command & the failure to meet the
 presence of the %1 variable (you JUST typed BATSEEK, nothing else).
actual pgm:The following batch file can be used to establish a pass-
 word for running a program. The batch file is named PASSWRD.BAT and
 calls up a fictitious program name MY.COM:
    {NOTE:[fghev xcvw] == our notes & 0) represents EDLIN line #s}
 1) ECHO OFF   [allows messaging in file w/next cmnd.
 2) CLS   to clear the ECHO OFF from the screen]
 3) IF %1 == ABC GOTO FINE [this is the label][IF TRUE,JUMP occurs]
 4) ECHO BAD PASSWORD--ENDING [msg./appears if NO ABC,then NO JUMP]
 5) GOTO END [ALWAYS end EA.cmnd.sequencing with this cmnd/label]
 6) :FINE [first label to wh/file JUMP occurs when statement TRUE]
 7) ECHO YOU'RE IDENTIFIED--STARTING CMD.SEQUENCE [msg/1st part:GOOD]
 8) MY   [the command portion of the :GOOD command sequence]
 9) GOTO END [remember, end each command sequence this way]
10) :END [the second JUMP label--remember labels direct traffic]
11) ECHO I'VE DONE MY JOB!--RETURNING TO CURRENT DRIVE [end msg.]
elucidation:Look at the response of the computer to various cmds.
 First a BAD password. At the prompt type: PASSWRD XYZ [xyz==%1]
 The computer tests for TRUE [it's NOT since %1 doesn't == ABC] and
 [goes to NEXT line] performs NEXT command, which is to display msg:
                        BAD PASSWORD--ENDING
 Now, a GOOD password. At the prompt type: PASSWRD ABC [now,abc==%1]
 The computer tests for TRUE [it IS since %1 == ABC] and performs the
 GOTO :FINE command and performs JUMP to label :FINE. P.C.then does
 what you told it to do after the label :FINE and performs the comm-
 and prompt "MY" after acknowledging the passwrod with a message. Af-
 ter MY performs you are returned to :END and sign off. Note the cmd.
 MY will not appear because echo is off and you didn't call for ECHO.
 The :END label displays the final message, but there is no require-
 ment that additional lines appear after end.

NOTES: Unlike programming languages, which allow many logical tests
 in an IF statement, the batch IF statement is limited to only the
 three named above. TO FURTHER EXPLAIN:
  >Condition 1:ERRORLEVEL is a number that indicates to DOS whether
  the last program run was succesful. A zero (0) indicates a good
  run, anything above zero indicates an error condition (only DOS
  commands BACKUP and RESTORE have an exit code). You probably WILL
  NOT have use for this feature often. There ARE commercial programs
  which make extensive use of ERRORLEVEL.These allow PC-User response
  and other features like alternate selection via Yes & No or selec-
  tion of numbers and/or letters (1,2,3,4 OR a,b,c,d, etc.).

```
  >Condition 2:String comparison is indicated by a double equal sign:
                        String1 == String22
 This compares the two strings. It is often used with parameters and
 markers to check for a particular entry. For example:
 IF %1 == 40  typing:"MODE C40" checks parameter one for 40 and, if
 if the condition is TRUE [%1 == 40] it changes the display to 40-
 columns wide. [SEE batch files on this disk for examples/study them]

   >Condition 3:The logical test for checking for the existence of a
 file has as its format: EXIST C:<filename.ext>
 You can use this test to check and see if a file is in C: drive
 (as in the example). EXIST B:<filename.ext> might be used to check
 to see if B: drive has a DOS disk in the drive. Explore other uses
 after examining files on this disk with "EXIST". Note: you cannot
 use pathnames for checking on a file's existence.

---------------- T  I  M  E  M  A  S  T  E  R --------------------
--------------B A T C H L R N  H E L P  S Y S T E M----------------
command: IF EXIST
use: The IF EXIST cmnd. allows CONFIRMATION of a file or condition.

how: Type:IF EXIST <filename.ext> or <condition> [then]  <cmnd.> O-R
          IF NOT EXIST <same as above>

example: IF EXIST %1 GOTO EXISTS
         ECHO    . . . T H E  F I L E  D O E S N ' T  E X I S T !
         :EXISTS
         ECHO  ...%1 IS AN EXISTING FILE IN THE CURRENT DIRECTORY!

 If you were to type:"EXIST SWELL.TXT" and it DID NOT EXIST the
 statement would be FALSE and the batch file would move to the next
 line, displaying: ...THE FILE DOES NOT EXIST! If you typed the same
 phrase, and the file WAS in the CURRENT directory the statement
 would be true and the batch file would JUMP to the label "EXISTS".
 It would then display the message: ...SWELL.TXT [that's the %1] IS
 AN EXISTING FILE IN THE CURRENT DIRECTORY! Using this basic shell
 you can create many IF EXIST batch processing files to make your
 work easier. FOR CROSS-REFERENCE SEE THE IF.HLP FILE!
NOTES: When <condition> is TRUE, DOS or batch <command> is performed.
 When <condition> is NOT TRUE,the <command> is IGNORED and the next
 line in the batch processing file is performed.
 The<condition> we are "testing" here, IF EXIST o-r NOT EXIST is ill-
 illustrated by these two examples:

          IF EXIST = True if and only if <filename> exists.

      IF NOT EXIST = True if and only if <condition> is false.

       Another example of IF EXIST (>Condition 3) is as follows:
   >Condition 3:The logical test for checking for the existence of a
 file has as its format: EXIST C:<filename.ext>
 You can use this test to check and see if a file is in C: drive
 (as in the example). EXIST B:<filename.ext> might be used to check
 to see if B: drive has a DOS disk in the drive. Explore other uses
 after examining files on this disk with "EXIST". Note: you cannot
 use pathnames for checking on a file's existence.

READER NOTE:The IF & IF EXIST cmnds are important to you. Review THIS
 file & IF.HLP,then s-t-u-d-y the "EXIST".BAT files.YOU'LL HAVE IT!
```

---------- T I M E M A S T E R ----------
----------B A T C H L R N H E L P S Y S T E M----------
command:MORE (with options)
use:MORE controls file screen output by limiting to 24 li. @ a time.
how:Type: TYPE <file.ext> | MORE
THE MORE COMMAND GOES IN A BATCH FILE AFTER THE
NAME OF THE FILE (TYPE MY.DOC | MORE) [for example]

N-O-T-E:MORE is a DOS filter (like FIND and SORT). It can be used with the Type command or with other commands,where you want to view 24 lines or one screen at a time. To advance you hit any key (hitting the space bar or return will NOT leave a character.[Desireable if you are using PrtSc]). See many of the Batch files on this Disk to see some interesting applications of MORE when combined with the other DOS filters in batch processing files. Many keystrokes and much time can be saved this way!

----------T I M E M A S T E R----------
----------B A T C H L R N H E L P S Y S T E M----------
commandS:Various PARAMETERS used in commands, including batch cmds.
use:To modify or otherwise change the effect of the command.

how:Type <command> <parameter> or use parameters in batch files.

examples:There are many batch files on this disk you can study to see how Parameters work. Also see examples included below.

explanation:Parameters are extra pieces of information that you type after many DOS cmnds.or in batch file instructions.DIR B:/W,for example, illustrates modification of the DOS command DIR. By adding the parameters B: and /W you are modifying basic operation of the command. You pass parameters to batch files the same way--by typing the information after the batch command, but BEFORE typing the balance of the information. This process is called "testing". This will seem complicated at first, but once you get the hang of it you will wonder why you didn't know about batch processing (replaceable) parameters before. Parameters may be used any place in a batch file they are needed to "test" for the presence of something you typed, (or DIDN'T TYPE) at the prompt (it WILL get clearer!).
The secret to understanding batch file parameters is to slowly work into the subject. The batch processing file uses parameters in several ways as it is being run and this (at first) makes comprehension difficult. "Markers" are used within the batch file as "signals" as to wh/parameter goes where. Markers are comprised of a percent sign (%) and a single digit between 0 and 9 (that's ten markers in use at any one time--remember that zero is a number). The %0 is ALWAYS the NAME OF THE BATCH PROCESSING FILE! As if it wasn't confusing enough already, many manuals refer to markers as simply "variables". The name, obviously, is not as important as the concept. To cover this, however, we also have a file called:VAR.HLP. It covers the fact that THERE IS a difference between parameters and variables.

examples: Assume that a batch file named COLORGO.BAT is on the current drive. This file contains ONLY a single command:
ECHO %0 %1 %2 %3 [ECHO shows messages on the screen].
If, at the DOS prompt, you typed: COLORGO Red Blue Green THAT cmd. would recognize that FOUR parameters were passed to the batch file. The batch file via the ECHO would display them on the screen in the order received (0=colorgo{name of .BAT file}; %1=red; %2=blue & %3=green) in the form you typed in. The parameters and markers were

related,(to restate),as follows... ECHO COLORGO Red Blue Green
                                    %0   %1 %2   %3

TO EXPLAIN REPLACEABLE PARAMETERS ANOTHER WAY:

You can include dummy parameters within your batch file which DOS will replace with values you supply on the command line when executing the file. Using replaceable parameters allows you to specify DIFFERENT sets of data every time you run the batch file. You can use up to 10 parameters; if more are needed, the shift command (SEE SHIFT.HLP)can be used to extend the available set. Note that OTHER areas in the batch file can recognize the %X parameters as well.For example, you might want a message: ECHO ...WORK COMPLETE ON %1 the batch file would repeat the message but replace %1 with the name you typed on the command or prompt line. Examine the .BAT files on this disk to see many examples of how this is accomplished. Another example of replaceable parameters is shown below:

```
FINDIT B: ROUND BLUE
```

When you type in a command, the command itself (FINDIT) is parameter %0. The parameter following the COMMAND (B:) is %1, and so on. In the example, therefore, ROUND=%1 and BLUE=%2 (a % for each word) All parameters following the cmd.take on succesive parameter values.

Here is another way to illustrate parameters (the loop file):

```
LOOP MYDEMO.DOC
REM - LOOP BATCH FILE FOR %1
TYPE %1
.
.           (MYDEMO.DOC {%!} is displayed)
.
CLS
LOOP
REM - LOOP BATCH FILE FOR %1
.
. (FILE IS DISPLAYED again)
.
FILE WILL KEEP GOING UNTIL YOU CTRL C OR CTRL-BREAK
```

While not particularly useful,this batch file serves to demonstrate replaceable parameters. There are a few ways in which such a file (looping batch file) like the one above could be used. It could continuously run a software demonstration package or graphics animation program with a loop. You could also test the integrity of hardware by running a continuous test program over and over until you stopped it.

Here's an experiment with replaceable parameters you can try:

```
(Create the following file)
COPY CON:SEEFILE.BAT
TYPE %1
TYPE %2
TYPE %3
<F6> <RETURN>
```

This batch file will display,using TYPE,the files that you specify as the first,second, and third parameters on the command line. Each filename.ext type would be typed in the order requested, on the CRT. screen. You could also use *.* (wildcards) as %1 and it would display a whole group of files (or *.DOC to display THOSE files). You could also use *.DOC as %1; *.TXT as %2, etc. When you use MULTIPLE PARAMETERS in a batch file they MUST be separated by spaces. To ill-

ustrate:Often you have repetitive tasks which are quite similar,perhaps differing only by the file being processed. A parameter is used to provide a file name to the .BAT file. This way, a batch file can immediately perform a task for you without having to be specially edited to match your current needs. For example, INFORM.BAT :

On monday:
  INFORM THUR.DAT FRI.DAT SAT.DAT SUN.DAT
On thursday:
  INFORM MON.DAT TUE.DAT WED.DAT
Up to 10 parameters may be used. Our example, INFORM.BAT would have a command line in its body. It could be COPY %1 + %2 + %3 + %4=A:REPORT (which would copy the files into ONE called REPORT which could then be renamed whatever you wanted). Or you could give a PRINT cmd. such as: PRINT %1 %2 %3 %4. There is no limit to your instructions.

REVIEW THE .BAT FILES ON THIS DISK WITH PARAMETERS

There may be times when you want to create an application program and run it with different sets of data. This data may be stored in various DOS files. Let's illustrate this & then examine our work. For example, when you type the command line COPY CON DOFILE.BAT, the next lines you type are copied from the console to a file named DOFILE.BAT on the default drive:

```
COPY CON DOFILE.BAT
COPY %1.INF+%2.INF=%2.PRN
TYPE %2.PRN
PRINT %2.PRN
```

Now press <F6> or <ctrl/z> and then press <return>. DOS responds:

"1 file(s) copied"

The file DOFILE.BAT, which consists of three commands, now resides on the disk in the default drive. It stands ready for variables! The dummy parameters %1 and %2 are replaced sequentially by the parameters you supply when you execute the file. The dummy parameter %0 is always replaced by the drive designator, if specified, and the filename of the batch file (for example:DOFILE).Now let's do it! To execute the batch file DOFILE.BAT and to specify the parameters that will replace the dummy param.(markers),you must type the batch file name(without its extension) followed by the parameters you want DOS to substitute for %1, %2, etc. Let's look at what happens:

(Remember that the file DOFILE.BAT consists of 3 lines):

```
COPY %1.INF+%2.INF=%2.PRN
TYPE %2.PRN
PRINT %2.PRN
```

To execute the batch processing file, DOFILE you might type:

```
DOFILE A:DATA1 B:DATA2
```

DOFILE is substituted for %0, A:DATA1 for %1, and B:DATA2 for %2.

(notice how the drives can be inserted)

The result is the same as if you had typed each of the commands in DOFILE with their parameters, as follows:

```
COPY A:DATA1.INF+B:DATA2.INF=DATA2.PRN [COPY DATA1&DATA2=DATA2.PRN]
TYPE B:DATA2.PRN    [DISPLAYING THE COMBINED FILE]
PRINT B:DATA2.PRN   [PRINT THE COMBINED FILE]
```

The following illustrates how DOS replaces each of the parameters:

BATCH FILENAME	PARAMETER1 (%0) (DOFILE)	PARAMETER2 (%1) (DATA1)	PARAMETER3 (%2) (DATA2)
DOFILE	DOFILE.BAT	DATA1.INF	DATA2.INF
			DATA2.PRN

1. Up to 10 dummy parameters (%0-%9) can be specified. Refer to the DOS command SHIFT if you wish to specify more than 10 parameters.
2. If you use the percent sign as part of a FILE NAME within a batch file, you must type it twice. For example, to specify the file XYZ%.BAT, you must type it as XYZ%%.BAT in the batch file.

---------- T I M E M A S T E R ----------
----------B A T C H L R N H E L P S Y S T E M----------
command: PAUSE
use: To pass to a user control of a batch file while processing.

how:Type:(in a batch file) PAUSE <Message, if any>

example:Virtually every >BAT file on this disk has a PAUSE command. TIMEMASTER displays messages on a separate line so we can insert different colors for the message and the PAUSE instruction in the color versions of these files.

NOTES: The basic function of PAUSE is to stop the execution of the batch file until you press a key. This can allow you to perform a necessary task -- like perhaps changing a disk or verifying that a particular disk configuration is in place, in order to avoid errors as the remaining parts of the batch file are executed. Pause will optionally display a message.
The message will show, followed by the DOS message:
Strike any key when ready...
THE PAUSE SUBCOMMAND IN BATCH PROCESSING FILES...
The PAUSE subcommand suspends execution of the batch file temporarily and displays the message "Strike any key when ready..." You can display messages in a batch file with the ECHO subcommand and then pause so the reader can read the message. These messages must precede the PAUSE command. A common method used in batch files is to display a message before a major command is issued, thus giving the user a chance to break out of the batch file. Example:

```
ECHO DRIVE C IS ABOUT TO BE ERASED...
ECHO Press Ctrl-Break to stop or
PAUSE
```

The screen display would appear as follows:

```
DRIVE C IS ABOUT TO BE ERASED...
Press Ctrl-Break to stop or  [also Ctrl C]
Strike any key when ready...
```

One of the most useful features of PAUSE is to allow a user to change diskettes between commands of a batch file.
You can use the PAUSE command to segment a batch file into segments that can be stopped at any appropriate point. <comment> is optional and should be entered on the same line as PAUSE. <comment> is used to prompt -- with a meaningful message -- the batch file user to take some action when the file pauses. The <comment> is displayed before the "Strike any key when ready..." message.

---------- T I M E M A S T E R ----------
----------B A T C H L R N H E L P S Y S T E M----------
command: REMark
use: The REM command displays on the CRT screen any text that is on the same line as the REM cmd.when a batch processing file is running
how:Type:(in a batch processing file) REM <comment or message>
actual batch file:

```
REM This file checks new disks
REM It is named MAKEDISK.BAT
PAUSE Insert new disk in drive B:
FORMAT B:/S
CHKDSK B:
```

NOTES: REMark can be used to send messages to the screen or simply to document some part of your batch file's operation. REM won't work with ECHO! We prefer REM ONLY for short files. The computer operator(it won't always be you)will want to know what is happening.Also, after some time,you may forget what a complicated set of commands actually does. REM messages can be up to 123 characters long,but DO keep the width of the screen in mind. This way you can control the display and make it even more meaningful. CENTER remarks when you can. ALL of the foregoing of course also applies to ECHO messages. If a line in a batch file starts with one or more periods (.) it is treated as a remark (in DOS 2.xx, DOS 3.x won't allow this). THUS REM This is a test a-n-d . This is a test are treated equally.

---------- T I M E M A S T E R ----------

----------B A T C H L R N H E L P S Y S T E M----------

command: SHIFT

use: The SHIFT command lets your batch file access more than 10 replaceable parameters.

example:The SHIFT command is illustrated by this example:

```
     [the lines below are as they would appear in a batch file]
     If %0 = "bak"
     %1 = "mine"
     %2 = "yours"
     %3...%9 are empty
     then a SHIFT (command) results in the following:
     %0 = "bak"
     %1 = "yours"
     %2...%9 are empty  ["mine" has been "shifted" out]
```

If you are still confused review the PARAM.HLP file.

NOTES:DOS allows 10 parameters (%0-%9) to be used by a batch file. SHIFT allows you to move more parameters into a queue, discarding the first parameter. When you are listing parameters at the prompt, additional parameters (more than 10) should be on the same command line in order to be shifted into the queue.The single cmd.is SHIFT. When that subcommand is encountered, all parameter/marker pairings are shifted one to the left. Whatever was assigned to %0 is lost, the contents of %1 are moved to %0, %2 moves to %1... %9 moves to %8 and a new parameter from the command line is moved into %9. While this brings in a new parameter, all have shifted and you must anticipate the effect on your batch file "program".

The effect of SHIFT:

```
 %0   %1   %2   %3   %4   %5   %6   %7   %8   %9
  |       Remember, this command seems very simple, but its effects
  |       are far ranging and the logical consequence of mis-matching
  |       parameters and markers can be disastrous!
  |
```

Lost

MORE ABOUT THE SHIFT SUBCOMMAND:

The SHIFT subcommand is made to be used with replaceable parameters. In the PARAM.HLP file there is an example for you to create called SEEFILE.BAT which shows you how to display three files you call up. What if you wanted to display four files? You would have to rewrite the batch file to include another TYPE command, or you could use the SHIFT command for more than 10 parameters in a file. The following batch file uses the GOTO and SHIFT command to ECHO any number of parameters typed on the command line.

(you should create this file)

```
      COPY CON:REVEAL.BAT
```

```
       ECHO OFF
       CLS
       :LOOP
       CLS
       TYPE %1 | MORE
       SHIFT
       GOTO LOOP
       <F6> or <Ctrl Z> <Return>
 To execute type: REVEAL <FILENAME.EXT> <FILENAME.EXT>,ETC.,ETC.

 CAUTION:When this file runs out of parameters you will get a message
 "invalid number of parameters". It will then KEEP REPEATING until
 YOU STOP IT with a Ctrl C or Ctrl-Break! The file can only "read"
 the number of parameters you put on the prompt or command line.
 NOTE: the MORE command is to assure that if a text file (that is
 the only kind you can display with TYPE) is more than 24 lines, a
 pause will be inserted. Wildcards cannot be used with the command.

---------------- T  I  M  E  M  A  S  T  E  R --------------------
---------------B A T C H L R N  H E L P  S Y S T E M----------------
command:SORT (with options)
use:SORT puts lines from a specified file into a particular order.
how:Type: "SORT < file.ext" = Sorted file will be displayed on scrn.
          "SORT /R <file.ext"=REVERSE sorted file to screen.
          "SORT+n < file.ext"=Sort by character in col.n(number).*
          "SORT < file.ext > sortedfile.ext=Sorts file.ext,then puts
              data in sortedfile.ext.
          "SORT < file.ext >> sortedfile.ext=Sorts file.ext, then
              APPENDS sorted data to EXISTING sortedfile.ext.
          "SORT < CON > file.ext=Sorts as you type data on keyboard,
              then places sorted data into file.ext.
          "SORT /R+n < file.ext"=Sort in reverse by col.n.*
      ANY OF THE ABOVE COMMANDS CAN BE INSERTED IN A BATCH FILE

  N-O-T-E:Since SORT is a filter it gets its data via redirection < .
  *column n=1-80 [look at the ASCII text, count from the left].Sort
  is by ASCII preference [#$%&'()*+0 1-9 :;<=>?@ A-Z [\]^ a-z{|},etc]
  Tips:DIR|SORT > PRN [prints DIR] & DIR|SORT > DIRECT.LST [stores
  in a file of your choice].
---------------T  I  M  E  M  A  S  T  E  R-----------------
---------------B A T C H L R N  H E L P  S Y S T E M----------------
command: CTRL C (MS-DOS)  or CTRL-BREAK or  SCR LOCK (PC-DOS)

use: To stop a batch file from being processed.

how:Type:CTRL C as batch file is running and it will stop for your
 Yes or No confirmation as to discontinuance.

NOTES:When you stop a batch file,the message is:Abort batch job?(Y/N)
 If you press N for No the task the batch file is performing goes on.

---------------- T  I  M  E  M  A  S  T  E  R --------------------
---------------B A T C H L R N  H E L P  S Y S T E M----------------
command:%<name>% (within a batch processing file)

use:A way of passing specific data to a batch processing file.

how:Type: %<name>% to set the variable.

NOTES:Variables provide another way of passing specific data to a
```

batch processing file. Please make THIS distinction, there IS an important difference between variables and parameters. PARAMETERS are constantly changing.A variable,once declared,RETAINS its value until either you reset it or turn the PC off. Using the variable you can not only pass values to the system but to other .BAT files. A variable is a text string(a name)enclosed by % signs (%variable%) Values are given to variables with the DOS SET command. For example, if a batch file contains the statement: LINK %FILE% you can "set" %FILE% to ANOTHER name (which DOS will then recognize, having discarded FILE and replacing it with the new name). In the example,and to replace %FILE% you would type: SET FILE = DOFILE (DOFILE now is the variable).

---------- T I M E M A S T E R ----------

END END END END END END END END END END END END END END

DATA2.PRN

NOTES

1. Up to 10 dummy parameters (%0-%9) can be specified. Refer to the DOS command SHIFT if you wish to specify more than 10 parameters.
2. If you use the percent sign as part of a FILE NAME within a batch file, you must type it twice. For example, to specify the file XYZ%.BAT, you must type it as XYZ%%.BAT in the batch file.

---------- T I M E M A S T E R ----------

#
#
#

----------B A T C H L R N H E L P S Y S T E M----------

command: PAUSE
use: To pass to a user control of a batch file while processing.

how:Type:(in a batch file) PAUSE <Message, if any>

example:Virtually every >BAT file on this disk has a PAUSE command. TIMEMASTER displays messages on a separate line so we can insert different colors for the message and the PAUSE instruction in the color versions of these files.

NOTES: The basic function of PAUSE is to stop the execution of the batch file until you press a key. This can allow you to perform a necessary task -- like perhaps changing a disk or verifying that a particular disk configuration is in place, in order to avoid errors as the remaining parts of the batch file are executed. Pause will optionally display a message.

The message will show, followed by the DOS message:

Strike any key when ready...

THE PAUSE SUBCOMMAND IN BATCH PROCESSING FILES...

The PAUSE subcommand suspends execution of the batch file temporarily and displays the message "Strike any key when ready..." You can display messages in a batch file with the ECHO subcommand and then pause so the reader can read the message. These messages must precede the PAUSE command. A common method used in batch files is to display a message before a major command is issued, thus giving the user a chance to break out of the batch file. Example:

```
ECHO DRIVE C IS ABOUT TO BE ERASED...
ECHO Press Ctrl-Break to stop or
PAUSE
```

The screen display would appear as follows:

DRIVE C IS ABOUT TO BE ERASED...

```
               Press Ctrl-Break to stop or  [also Ctrl C]
               Strike any key when ready...
 One of the most useful features of PAUSE is to allow a user to
 change diskettes between commands of a batch file.
 You can use the PAUSE command to segment a batch file into segments
 that can be stopped at any appropriate point. <comment> is optional
 and should be entered on the same line as PAUSE. <comment> is used
 to prompt -- with a meaningful message -- the batch file user to
 take some action when the file pauses. The <comment> is displayed
 before the "Strike any key when ready..." message.
---------------- T  I  M  E  M  A  S  T  E  R --------------------
---------------B A T C H L R N  H E L P  S Y S T E M----------------
command: REMark
use: The REM command displays on the CRT screen any text that is on
 the same line as the REM cmd.when a batch processing file is running
how:Type:(in a batch processing file) REM <comment or message>
actual batch file: REM This file checks new disks
                   REM It is named MAKEDISK.BAT
                   PAUSE Insert new disk in drive B:
                   FORMAT B:/S
                   CHKDSK B:
NOTES: REMark can be used to send messages to the screen or simply
 to document some part of your batch file's operation. REM won't work
 with ECHO! We prefer REM ONLY for short files. The computer opera-
 tor(it won't always be you)will want to know what is happening.Also,
 after some time,you may forget what a complicated set of commands
 actually does. REM messages can be up to 123 characters long,but DO
 keep the width of the screen in mind. This way you can control the
 display and make it even more meaningful. CENTER remarks when you
 can. ALL of the foregoing of course also applies to ECHO messages.
 If a line in a batch file starts with one or more periods (.) it is
 treated as a remark (in DOS 2.xx, DOS 3.x won't allow this). THUS
 REM This is a test  a-n-d  . This is a test are treated equally.
---------------- T  I  M  E  M  A  S  T  E  R --------------------
---------------B A T C H L R N  H E L P  S Y S T E M----------------
command: SHIFT
use: The SHIFT command lets your batch file access more than 10
 replaceable parameters.

example:The SHIFT command is illustrated by this example:
     [the lines below are as they would appear in a batch file]
     If %0 = "bak"
     %1 = "mine"
     %2 = "yours"
     %3...%9 are empty
     then a SHIFT (command) results in the following:
     %0 = "bak"
     %1 = "yours"
     %2...%9 are empty  ["mine" has been "shifted" out]
 If you are still confused review the PARAM.HLP file.

NOTES:DOS allows 10 parameters (%0-%9) to be used by a batch file.
 SHIFT allows you to move more parameters into a queue, discarding
 the first parameter. When you are listing parameters at the prompt,
 additional parameters (more than 10) should be on the same command
 line in order to be shifted into the queue.The single cmd.is SHIFT.
 When that subcommand is encountered, all parameter/marker pairings
 are shifted one to the left. Whatever was assigned to %0 is lost,
 the contents of %1 are moved to %0, %2 moves to %1... %9 moves to
 %8 and a new parameter from the command line is moved into %9.
```

While this brings in a new parameter, all have shifted and you must anticipate the effect on your batch file "program".
The effect of SHIFT:

```
 %0   %1   %2   %3   %4   %5   %6   %7   %8   %9
  |       Remember, this command seems very simple, but its effects
  |       are far ranging and the logical consequence of mis-matching
  |       parameters and markers can be disastrous!
  |
```

Lost

MORE ABOUT THE SHIFT SUBCOMMAND:

The SHIFT subcommand is made to be used with replaceable parameters. In the PARAM.HLP file there is an example for you to create called SEEFILE.BAT which shows you how to display three files you call up. What if you wanted to display four files? You would have to rewrite the batch file to include another TYPE command, or you could use the SHIFT command for more than 10 parameters in a file. The following batch file uses the GOTO and SHIFT command to ECHO any number of parameters typed on the command line.

(you should create this file)

```
COPY CON:REVEAL.BAT
ECHO OFF
CLS
:LOOP
CLS
TYPE %1 | MORE
SHIFT
GOTO LOOP
<F6> or <Ctrl Z> <Return>
```

To execute type: REVEAL <FILENAME.EXT> <FILENAME.EXT>,ETC.,ETC.

CAUTION:When this file runs out of parameters you will get a message "invalid number of parameters". It will then KEEP REPEATING until YOU STOP IT with a Ctrl C or Ctrl-Break! The file can only "read" the number of parameters you put on the prompt or command line. NOTE: the MORE command is to assure that if a text file (that is the only kind you can display with TYPE) is more than 24 lines, a pause will be inserted. Wildcards cannot be used with the command.

---------- T I M E M A S T E R ----------
----------B A T C H L R N H E L P S Y S T E M----------

command:SORT (with options)
use:SORT puts lines from a specified file into a particular order.
how:Type: "SORT < file.ext" = Sorted file will be displayed on scrn.
"SORT /R <file.ext"=REVERSE sorted file to screen.
"SORT+n < file.ext"=Sort by character in col.n(number).*
"SORT < file.ext > sortedfile.ext=Sorts file.ext,then puts data in sortedfile.ext.
"SORT < file.ext >> sortedfile.ext=Sorts file.ext, then APPENDS sorted data to EXISTING sortedfile.ext.
"SORT < CON > file.ext=Sorts as you type data on keyboard, then places sorted data into file.ext.
"SORT /R+n < file.ext"=Sort in reverse by col.n.*
ANY OF THE ABOVE COMMANDS CAN BE INSERTED IN A BATCH FILE

N-O-T-E:Since SORT is a filter it gets its data via redirection < . *column n=1-80 [look at the ASCII text, count from the left].Sort is by ASCII preference [#$%&'()*+0 1-9 :;<=>?@ A-Z [\]^ a-z{|},etc]
Tips:DIR|SORT > PRN [prints DIR] & DIR|SORT > DIRECT.LST [stores in a file of your choice].

|----------T I M E M A S T E R----------|
#
#
#
#
#
#
|----------B A T C H L R N H E L P S Y S T E M----------|
command: CTRL C (MS-DOS) or CTRL-BREAK or SCR LOCK (PC-DOS)

use: To stop a batch file from being processed.

how:Type:CTRL C as batch file is running and it will stop for your Yes or No confirmation as to discontinuance.

NOTES:When you stop a batch file,the message is:Abort batch job?(Y/N) If you press N for No the task the batch file is performing goes on.

|---------- T I M E M A S T E R ----------|
#
|----------B A T C H L R N H E L P S Y S T E M----------|
command:%<name>% (within a batch processing file)

use:A way of passing specific data to a batch processing file.

how:Type: %<name>% to set the variable.

NOTES:Variables provide another way of passing specific data to a batch processing file. Please make THIS distinction, there IS an important difference between variables and parameters. PARAMETERS are constantly changing.A variable,once declared,RETAINS its value until either you reset it or turn the PC off. Using the variable you can not only pass values to the system but to other .BAT files. A variable is a text string(a name)enclosed by % signs (%variable%) Values are given to variables with the DOS SET command. For example, if a batch file contains the statement: LINK %FILE% you can "set" %FILE% to ANOTHER name (which DOS will then recognize, having discarded FILE and replacing it with the new name). In the example,and to replace %FILE% you would type: SET FILE = DOFILE (DOFILE now is the variable).

|---------- T I M E M A S T E R ----------|
→