WECHO.COM V 1.33

THE EXTENDED ECHO FOR YOUR BATCH PROGRAMS

IMPORTANT:

If you want to use the ENGLISH version of Wecho,
please copy WECHO_E.COM to your computer and
rename it into WECHO.COM then.

CONTENTS

Distribution of the program

System requirements

WECHO.COM V1.33 2002 The extended echo command for batch programs
---
(C) Claus-Juergen Claussen, Riedwiesenweg 10, 69181 Leimen, Germany
    claussen.leimen@t-online.de

```
Test the following commands:

WECHO /?    or   WECHO     = help screen
WECHO #                    = makes a new line
WECHO :It is #J:#I #Um on #W, #M #X, #Y.
WECHO :It is #j:#i #um on #w, #m #x, #y.> filename.xxx

Replacement parameters always start with #

#Y year    #M month   #W day of week  #X day of month
#H hour    #N month (as figure)     #T second (one figure)
           #I minute  #S second     #Z 1/100 second

In Batch files also % parameters will work:
WECHO :text %3 text #H %NAME% text #M #T >> filename.xxx
```

WECHO.COM has a considerably extended functionality compared to the batch command ECHO. The program employs replacement parameters which always start with #. % parameters used by DOS will also function as well as the character > used for redirection of texts into files. Many functions of WECHO also work in DOS boxes of WINDOWS operating systems.

Example for an invocation:

```
WECHO Comment 1:It is #J:#I #Um#;Comment 2
```

Meanings of the parts:

Comment 1:	The command line is not evaluated before the first colon. Thus, a comment can be placed here.
It is...	Text to be output including replacement parameters
#;Comment 2	After #; the command line is no longer evaluated. Thus, another comment can be placed here.

Any number of # replacement parameters can be inserted into a command line. They cause special text outputs or other functions.

```
Date and time (#$. as of version 1.33)
--------------------------------------
#W  Day of week, e.g. Tuesday        #H  Hour, e.g. 17 or 3
#$W Day of week, e.g. TUE            #$H Hour, e.g. 17 or 03
#X  Day of month, e.g. 25 or 7       #I  Minute, e.g. 25 or 05
#$X Day of month, e.g. 25 or 07      #$I Minute, e.g. 25 or 5
#M  Month, e.g. September            #S  Second, e.g. 13 or 05
#$M Month, e.g. SEP                  #T  Second, e.g. 13 or 5
#N  Month, e.g. 12 or 9
#$N Month, e.g. 12 or 09
#Y  Year, e.g. 1994 or 2005          #Z  1/100 sec. e.g. 78 or 03
#$Y Year, e.g. 94 or 05              #$Z 1/100 sec. e.g. 78 or 3

#J  Hour 1...12 (British/American time), as of version 1.1
#$J Hour 01...12 (British/American time), as of version 1.33
#U  Outputs a or p for am, a.m., pm, p.m., as of version 1.1
#$U Outputs A or P for AM, A.M., PM, P.M., as of version 1.33
#G  Date and time completely in short form

Special characters
------------------
#00 to #FF   Hex value for a character to be output,
             for example: for characters reserved by DOS

             #3E >   #7C |   #1B <ESC>   #0C FF  (Form Feed)
             #3C <   #25 %   #09 <TAB>   #07 BEL (Bell)
             #26 &   #34 "   #5E ^       #40 @
             #0D CR alone    #0A LF alone

#V New line (CR+LF)
#P No new line (CR+LF) at the end of the output line
## outputs # once

Switching and diagnosis functions
---------------------------------
#_G turns CapsLock to ON    (no function in Win32)
#_H turns CapsLock to OFF   (no function in Win32)
#_N turns NumLock to ON     (no function in Win32)
#_O turns NumLock to OFF    (no function in Win32)
#_S turns ScrollLock to ON  (no function in Win32)
#_T turns ScrollLock to OFF (no function in Win32)
#_A turns CapsLock, NumLock and ScrollLock to OFF (no function in Win32)
#_\ outputs the current drive (as of Version 1.32)
#\  outputs the current directory
#-  deletes the keyboard buffer
#;  the command line is no longer evaluated
    (used for comments which are not to be output)

#=M Mouse check, with comment, with return code
    Code 0  Mouse is ready for use
    Code 1  Mouse was not found
    Code 2  Mouse driver was not found
#=N like #=M with return code, but without comment

#=P Printer test LPT1: with comment, with return code
    Code 0  Printer is ready for use
    Code 1  Printer is not ready for use or does not exist
#=Q like #=P  but for LPT2:
#=R like #=P  but for LPT3:
#=T Test LPT1: without comment, with return code
#=U Test LPT2: without comment, with return code
#=V Test LPT3: without comment, with return code
```

```
Creation of return codes (errorlevels)
--------------------------------------
Return codes can be produced at any position of the command line.
If more than one return code are produced, the last one will be
saved and can be evaluated by IF ERRORLEVEL then.

Setting a return code: #R00 to #RFF (hex value causes the code of 0...255).
Return code from the day of week (0...6, 0=Sunday):   #RW
            from the day of month (1...31):           #RX
            from the month (1...12):                  #RM
            from the year (80...199 = 1980...2099):   #RY
            from the hour (0...23):                   #RH
            from the minute (0...59):                 #RI
            from the second (0...59):                 #RT
            from the 1/100 second (0...99):           #RZ
Delete keyboard buffer and create return code:        #R-
            Code 0 = buffer was already empty
            Code 1 = buffer contained character(s)

Create a return code from a keyboard call:
#K waits for a keystroke (1 character) and creates a return code then:

   CTRL-A to CTRL-Z  1...26 (CR = 13, TAB = 9, BS = 8, ESC = 27)
   Space bar to Z (see ASCII table) 32...90  (a...z is equal to A...Z)

   ä,Ä = 142  <F1> to <F10> = 101 to 110
   ö,Ö = 153  Ins  = 182  CursorUp = 172  PgUp = 173  CtrlPgUp = 232
   ü,Ü = 154  Del  = 183  CursorDn = 180  PgDn = 181  CtrlPgDn = 218
   CtrlCurLeft  = 215     CurLeft  = 175  Home = 171  CtrlHome = 219
   CtrlCurRight = 216     CurRight = 177  End  = 179  CtrlEnd  = 217

#L has the same function as #K, but the character is displayed if
printable

Remark: #K and #L clear the keyboard buffer first, to ensure that
a real keyboard call is carried out, and to prevent that a keyboard
entry made before has no effect.

#RL Displays a return code as a number (while the program is running)
    Example: WECHO :#RFE#RL  (FE means 254 decimal, thus the return
                              code 254 is created and displayed as 254)

#RK Displays a return code as a letter (while the program is running)
#RG if return code = 1, CapsLock = ON    } Interrogation of CapsLock,
#RN if return code = 1, NumLock  = ON    } NumLock and
#RS if return code = 1, ScrollLock = ON  } ScrollLock
#R\ Return code of the current drive (A = 1, B = 2 and so on)
```

Creation of tones (and delay times)

#[xxx] For xxx certain characters are to be inserted (at least one) [means start of music,] means end of music. The characters between [and] stand for the following:

Set the length of tone: 1 period = 1/18.2 s = ca. 50 ms)

0 = 1 period	5 = 8 periods	{ = 48 periods	` = 256 per.		
1 = 2 periods	6 = 12 periods	} = 64 periods			
2 = 3 periods	7 = 16 periods	~ = 96 periods			
3 = 4 periods	8 = 24 periods	\ = 128 periods			
4 = 6 periods	9 = 32 periods	^ = 192 periods (not Win32)			

If the length of tone is not specified, the default setting 5 is active (= 8 periods).

Set the frequency of tone:
Space or _ (underscore): 0 Hz (delay time without tone)

	!	77.782 Hz	A	220.00 Hz	S	622.25 Hz	k 1760.0 Hz
(not Win32)	"	82.401 Hz	B	233.08 Hz	T	659.26 Hz	l 1864.7 Hz
	$	87.307 Hz	C	246.94 Hz	U	698.46 Hz	m 1975.5 Hz
(not Win32)	&	92.499 Hz	D	261.63 Hz	V	739.99 Hz	n 2093.0 Hz
	'	97.999 Hz	E	277.18 Hz	W	783.99 Hz	o 2217.5 Hz
	(	103.826 Hz	F	293.66 Hz	X	830.61 Hz	p 2349.3 Hz
	)	110.000 Hz	G	311.13 Hz	Y	880.00 Hz	q 2489.0 Hz
	*	116.541 Hz	H	329.63 Hz	Z	932.30 Hz	r 2637.0 Hz
	+	123.471 Hz	I	349.23 Hz	a	987.80 Hz	s 2793.8 Hz
	,	130.813 Hz	J	369.99 Hz	b	1046.5 Hz	t 2960.0 Hz
	-	138.591 Hz	K	392.00 Hz	c	1108.7 Hz	u 3136.0 Hz
	.	146.832 Hz	L	415.30 Hz	d	1174.7 Hz	v 3322.4 Hz
	/	155.563 Hz	M	440.00 Hz	e	1244.5 Hz	w 3520.0 Hz
	:	164.814 Hz	N	466.16 Hz	f	1318.5 Hz	x 3729.3 Hz
	;	174.614 Hz	O	493.88 Hz	g	1396.9 Hz	y 3951.1 Hz
	=	184.997 Hz	P	523.25 Hz	h	1480.0 Hz	z 4186.0 Hz
	?	195.998 Hz	Q	554.37 Hz	i	1568.0 Hz	
	@	207.652 Hz	R	587.33 Hz	j	1661.2 Hz	

Example for some tones with pause: WECHO :#[4ACE5BD2_3F]
Example for a delay time (ca. 1 s) without any tone: WECHO :#[8_]
Please note that the characters between [and] are case-sensitive (the upper-case letters cause other sounds than the lower case ones do, see the table above).

Running music (notes and pauses) can be aborted by a keystroke of any key. In case of an termination by a keystroke a return code is created (same code as with #K).
Remark: #[xxx] clears the keyboard buffer first, to ensure that a keyboard entry made before does not cause an unintended termination.

Internal text modules (see also the example NAME.BAT)

#Qx inserts an internal text module,
x stands for any character from a to z
#QA @ECHO OFF CR/LF SET NAME=
#QB @ECHO OFF CR/LF
#QC IF ERRORLEVEL
#QD IF NOT ERRORLEVEL
#QE to #QZ are not defined yet

Call upon an entry of a character string

#:x x stands for the ciphers 0 to 9 and for the letters a to z
as well. This function waits for a string entered via the
keyboard and terminated with <Enter>. The maximum length
of the string is 128 characters. The string is then stored
in the following file:

C:\WECHO\WECHO_x.BAT (x=0...9, a...z)

The directory C:\WECHO must exist.
The last two characters in the file are CR/LF. If this file
is executable as a batch program, it can be run from a just
running batch program using the CALL command.

Insertion of a character string from a file

#*x x stands for the ciphers 0 to 9 and for the letters a to z
as well. The contents of the file

C:\WECHO\WECHO_x.BAT (x=0...9, a...z)

is inserted. Possibly existing CTRL-Z characters are omitted.
The entire file is inserted (no limit of length).
TABs are converted into a suitable number of blanks.
In order to prevent problems, TABs and Ctrl-Z characters
should be used not at all.

During the insertion procedure, lower-case letters are con-
verted into upper-case letters. #*x works as of version 1.2.

#.x x stands for the ciphers 0 to 9 and for the letters a to z
as well. The contents of the file

C:\WECHO\WECHO_x.BAT (x=0...9, a...z)

is inserted. Possibly existing CTRL-Z characters are omitted.
The entire file is inserted (no limit of length).
TABs are converted into a suitable number of blanks.
In order to prevent problems, TABs and Ctrl-Z characters
should be used not at all. There is no lower-upper-case
conversion. #.x works as of version 1.2.

#+x x stands for the ciphers 0 to 9 and for the letters a to z
as well. The contents of the file

C:\WECHO\WECHO_x.BAT (x=0...9, a...z)

is inserted. Possibly existing CTRL-Z or CR/LF characters
are omitted. Only a maximum of the first 128 characters
of the file are inserted. TABs are converted into a suitable
number of blanks. In order to prevent problems, TABs and
Ctrl-Z characters should be used not at all. There is no
lower-upper-case conversion.
Since #+x removes all CR/LF characters, this function is
useful to cut these control characters from text strings.
Please keep in mind that the WECHO command always outputs
the CR/LF character pair at the end of the line, if it is
not suppressed by #p.

Standard directory for (temporary) WECHO files

#~x x stands for the letters a to z. For the functions
     #:x, #*x, #.x and #+x binding file names are predefined
     on the drive C:, e.g. C:\WECHO\WECHO_5.BAT. Thus the drive C:
     is predefined.
     Using the parameter #~x (x = A to Z), you can rename the drive
     allocation for the mentioned files to another one.
     This rename function is valid as long as it is not overwritten
     by a new definition or a new start of the WECHO command.
     Without renaming the drive, a new WECHO line always uses
     drive C:. The rename function enables you, for instance, to use
     a directory named E:\WECHO\.. for temporary files. Drive E: may
     advantageously be a RAM drive. In this case, the processing speed
     is increased and drive C: is relieved. If you have loops in your
     batch program or you have to access the drive repeatedly, you
     will notice the advantage. Do not forget to create the directory
     \WECHO\... on the RAM drive you intend to use with WECHO. This
     has to be performed with each boot procedure.
     Of course, you can use #~A in order to have temporary WECHO files
     on your disk drive A: (directory A:\WECHO\...).
     #~x works as of version 1.2.

     As of WECHO version 1.3 there is a new option. Now you can also
     predefine the drive for the WECHO files by using the system en-
     vironment. The entry is made by the SET command of DOS. If you
     desire that the drive E: should handle the WECHO files, please
     add the following lines to the contents of the AUTOEXEC.BAT
     file (rather than typing them on the DOS command level):

     SET WECHODRV=E
     MD E:\WECHO

     WECHO checks for the entry at WECHODRV and adopts the drive
     name specified there. If the drive is renamed within the WECHO
     command line using #~x, this new definition is valid.
     With the next call of WECHO, the definition under WECHODRV is
     valid again.

     Note: The DOS environment can become too small if used for a
     lot of SET entries. The size of the system environment can be
     enlarged to 1024 bytes by the following line entered into the
     CONFIG.SYS file (line is marked):

     DEVICE=C:\DOS\HIMEM.SYS
     DEVICE=C:\DOS\EMM386.EXE NOEMS
     DOS=HIGH,UMB
     LASTDRIVE=K
---> SHELL=C:\COMMAND.COM /P /E:1024
     COUNTRY=049,437,C:\DOS\COUNTRY.SYS
     DEVICEHIGH=C:\DOS\ANSI.SYS
     BREAK=OFF
     FILES=50
     BUFFERS=40,8
     STACKS=18,512
     DEVICEHIGH=C:\DOS\RAMDRIVE.SYS 44
     DEVICE=C:\DOS\RAMDRIVE.SYS 256 512 512 /E

```
Calculations with WECHO
-----------------------

#{......} Calculator (see examples),    works as of V 1.2
                                      function extensions as of V 1.31
    #{....} provides a signed integer calculator with 32 bits in size.
    The result of calculation is sent to the screen. It can be re-
    directed into a file.

    WECHO :...#{ sign_1  operand_1  operator  sign_2  operand_2 }
    WECHO :...#{    -       45          /       -         32    }

    Examples:
    WECHO :#{-45*-32}   WECHO :#{-231+-56}   WECHO :..#{78?-127}...

    Rules:
    1. Positive signs may be omitted
    2. The equation may be "shaked up" by inserting blanks
    3. Brackets () and [] may be used
    4. The character } at the end may be replaced by =
    5. Exactly 2 operands have to be used
    6. Only one operator must be given e.g. + - * / \ _ ?
    7. Operator ? compares the operands,
                a return code is generated in this case
                    1 = Operand 1 is greater
                    2 = Operand 2 is greater
                    3 = Operand 1 equals operand 2
       Operator / divides operand 1 by operand 2, the result is
                rounded up or down
       Operator \ divides operand 1 by operand 2, the result is
                always rounded down (pure integer division)
       Operator _ divides operand 1 by operand 2, instead of the
                result the remainder is output (modulo function)
    8. Calculation number range for inputs and results:
       -2 147 483 647...0...+2 147 483 647 (32 bits signed integer)
    9. Return codes: 0 = Everything is o.k.     8 = Error message
                     5 = Result of division is rounded off
                         or has a remainder
   10. Error messages:
       INPUT-ERROR:  Inputs are undefined or operand is out of range
       OUTPUT-ERROR: Calculation result is out of range
       DIVIDE-BY-ZERO: It was tried to divide by zero
       If an error was detected, WECHO aborts the program execution.
```

#o0...#o9 Mouse menu (see also the example MENU.BAT) as of V 1.3

#o0...#o9 (mouse menu) places a mouse cursor onto the screen.
By clicking a screen character its ASCII code generates a return
code. The mouse menu subprogram then terminates.
Control, blank and graphic characters (ASCII code greater than 165)
are ignored if selected by clicking. The parameter following "o" must
be a cipher (0...9). This cipher determines the field, within which
the mouse cursor can be moved. Example: WECHO :#o4
     0 = Mouse cursor can move around the entire screen field
 1...9 = Number of lines, on which the mouse cursor can be moved.
         Here, the horizontal movement is limited to 10 characters.

The menu image, within which the mouse cursor is to be moved has to be
output to the screen by the ECHO or TYPE command. Test the mouse
cursor movement using the batch file MENU.BAT. This batch file takes
advantage of the batch program ERRLEVEL.BAT for displaying the
return code.
If you press the ESC key while the mouse cursor is on the screen,
the mouse menu subprogram is terminated and the return code 255
is created. The same code is generated if

 - the mouse or the mouse driver is missing
 - the "o" of #o... is not followed by a cipher.

#(...) Cursor functions as of V 1.33

Upon call of #(...) (cursor functions) the cursor can be changed
in its shape and/or its position on the screen.
Between the two parantheses the following can be inserted:

 #(ON) or #(N) Cursor is switched ON (VGA and higher)
 #(OFF) or #(F) Cursor is switched OFF (VGA and higher)
 #(VGA) or #(V) Default setting for VGA and higher (also for MCGA)
 #(CGA) or #(C) Default setting for CGA and higher
 #(MDA) or #(M) Default setting for MDA, HERCULES, EGA
 #(BIG) or #(B) Big cursor
 #(-)   Thin cursor
 #(+)   Thick cursor
 #(X46)   Moves the cursor to the 46th column (values from 1 to 80)
 #(Y17)   Moves the cursor to the 17th linee (values from 1 to 25)

 It is possible to use several parameters in one command, e.g.
 WECHO :#(OFF X53 Y12)Hallo#(ON BIG)

Examples for batch programs using WECHO:

```
@echo off
rem 255.BAT creates the file 255.BIN, which contains all characters
rem from 000 to 255 (00 to FFH)
WECHO :#00#01#02#03#04#05#06#07#08#09#0a#0b#0c#0d#0e#0f#p>255.bin
WECHO :#10#11#12#13#14#15#16#17#18#19#1a#1b#1c#1d#1e#1f#p>>255.bin
WECHO :#20#21#22#23#24#25#26#27#28#29#2a#2b#2c#2d#2e#2f#p>>255.bin
WECHO :#30#31#32#33#34#35#36#37#38#39#3a#3b#3c#3d#3e#3f#p>>255.bin
WECHO :#40#41#42#43#44#45#46#47#48#49#4a#4b#4c#4d#4e#4f#p>>255.bin
WECHO :#50#51#52#53#54#55#56#57#58#59#5a#5b#5c#5d#5e#5f#p>>255.bin
WECHO :#60#61#62#63#64#65#66#67#68#69#6a#6b#6c#6d#6e#6f#p>>255.bin
WECHO :#70#71#72#73#74#75#76#77#78#79#7a#7b#7c#7d#7e#7f#p>>255.bin
WECHO :#80#81#82#83#84#85#86#87#88#89#8a#8b#8c#8d#8e#8f#p>>255.bin
WECHO :#90#91#92#93#94#95#96#97#98#99#9a#9b#9c#9d#9e#9f#p>>255.bin
WECHO :#a0#a1#a2#a3#a4#a5#a6#a7#a8#a9#aa#ab#ac#ad#ae#af#p>>255.bin
WECHO :#b0#b1#b2#b3#b4#b5#b6#b7#b8#b9#ba#bb#bc#bd#be#bf#p>>255.bin
WECHO :#c0#c1#c2#c3#c4#c5#c6#c7#c8#c9#ca#cb#cc#cd#ce#cf#p>>255.bin
WECHO :#d0#d1#d2#d3#d4#d5#d6#d7#d8#d9#da#db#dc#dd#de#df#p>>255.bin
WECHO :#e0#e1#e2#e3#e4#e5#e6#e7#e8#e9#ea#eb#ec#ed#ee#ef#p>>255.bin
WECHO :#f0#f1#f2#f3#f4#f5#f6#f7#f8#f9#fa#fb#fc#fd#fe#ff#p>>255.bin
```

```
@echo off
rem NAME.BAT  Asks for your name and enters it under
rem           NAME=.... into the system environment.
rem           Look it up with the DOS command SET
if not exist c:\WECHO\nul md c:\WECHO
if exist C:\WECHO\xhelp.bat del C:\WECHO\xhelp.bat
if exist C:\WECHO\WECHO_0.bat del C:\WECHO\WECHO_0.bat
WECHO :Please type your name: #:0
WECHO :#QA#+0>C:\WECHO\xhelp.bat
call C:\WECHO\xhelp
if exist C:\WECHO\xhelp.bat del C:\WECHO\xhelp.bat
if exist C:\WECHO\WECHO_0.bat del C:\WECHO\WECHO_0.bat
echo Your name has been entered.
:ende
```

```
@echo off
rem YN.BAT Test the keyboard call (for one character)
WECHO :Please press a key: #L#v
goto start
:oben
WECHO :#K#p
:start
if errorlevel 13 if not errorlevel 14 goto crturn
if errorlevel 27 if not errorlevel 28 goto escape
if errorlevel 89 if not errorlevel 90 goto yes
if errorlevel 78 if not errorlevel 79 goto no
WECHO :#[4ACM]Only with Y, N, ESC or ENTER you can reach the target#p#0d
goto oben
:crturn
WECHO :#vENTER was it
goto ende
:escape
WECHO :#vYou were successful with ESC
goto ende
:yes
WECHO :#vY like YES was the solution
goto ende
:no
WECHO :#vWith N like NO you found the exit
goto ende
:ende


@echo off
rem ROUSE.BAT The right word at every time of the day
WECHO :#rh
if errorlevel 22 goto nacht
if errorlevel 18 goto abend
if errorlevel 12 goto tag
if errorlevel 10 goto vormit
if errorlevel 8  goto morgen
goto frueh
:nacht
WECHO :It is #j.#i #um. It is too late to sit at the computer.
goto ende
:abend
WECHO :In the evening at #j.#i #um it is the best time for working a lot.
goto ende
:tag
WECHO :At #j.#i #um Do not bother the computer! The sun is shining.
goto ende
:morgen
WECHO :It is #j.#i #um. Have you already woken up?
goto ende
:vormit
WECHO :It is #j.#i #um. Why are you so busy in the morning hours?
goto ende
:frueh
WECHO :It is #j.#i #um. I would say it is still night!
goto ende
:ende
WECHO Just a line feed
```

```
@echo off
rem CAP.BAT  This file takes %1 to %9 from the command line and
rem writes them under the names CP1 to CP9 to the system environment.
rem From here, the parameters can (converted into upper-case letters)
rem be used as %CP1% to %CP9%, e.g. for IF "CP1"=="/A".....
WECHO :#qbSET CP1=%1#vSET CP2=%2#vSET CP3=%3#vSET CP4=%4#vSET
CP5=%5>c:\WECHO\WECHO_A.bat
WECHO :SET CP6=%6#vSET CP7=%7#vSET CP8=%8#vSET CP9=%9>>c:\WECHO\WECHO_A.bat
WECHO :#*A#p>help.bat
call help
del help.bat


@echo off
REM MENU.BAT (Mouse menu on the screen)
rem ANSI.SYS must be installed in CONFIG.SYS. Otherwise the colours
rem and the screen arrangement will not work. The programs WECHO.COM
rem and ERRLEVEL.BAT must be in the PATH or in the current directory.

echo                ┌──────────────────────────────────────┐
WECHO :             │ #1b[46;30m Please select with the mouse #1b[0m │
echo                ├──────────────────────────────────────┤
WECHO :             │  #1b[41;37mYYY#1b[0m        YES                  │
WECHO :             │       #1b[42;30mNNN#1b[0m   NO                   │
WECHO :             │  #1b[43;37mQQQ#1b[0m        QUIT                 │
WECHO :             │       #1b[44;37mEEE#1b[0m   EXIT                 │
WECHO :             │  #1b[45;37mAAA#1b[0m        Drive A:             │
WECHO :             │       #1b[46;30mBBB#1b[0m   Drive B.             │
echo                └──────────────────────────────────────┘
WECHO :#o6
call errlevel
echo ┌──────────────────────────────────────────────────┐
WECHO :│      #1b[46;30m  Please select with the mouse  #1b[0m      │
echo ├──────────────────┬──────────────┬────────────────┤
echo │ YES / NO         │ Y N Q E      │ QUIT / EXIT    │
echo │ DRIVE A: B:      │  A B C D     │ DRIVE C: D:    │
echo └──────────────────┴──────────────┴────────────────┘
WECHO :#o2
call errlevel


Distribution of the program
---------------------------
The program WECHO.COM is freeware. It may be used by everyone
for non-commercial purposes free of charge. The author assumes
no warranty or responsibility for errors or consequences caused
by the use of the program. Everybody may distribute the program,
but only with this description included.
For commercial use please contact the author for an agreement.
The program TE.COM comes with this package. It is
used for finding out which key codes are generated when keys
are pressed. The batch utility ERRLEVEL.BAT may also be useful
for you. It is used for determining the return code after a
program has terminated.
ERL.BAT fulfills the same requirement, but does not send the
return code to the screen. Instead of that it enters the return
code into the system environment under the name ERL.
TE.COM and ERRLEVEL.BAT are Public Domain Software.
You can use them in every way.
Perhaps you will find the program RECKON of interest.
It is also freeware. Most functions are integrated in WECHO,
see WECHO :#{.....}.
```

System requirements

You need an IBM-compatible XT/AT computer in text mode
(everybody has it). Processor 8088/86 and up. DOS 3.30 and up.
It is necessary to load ANSI.SYS. The normal computer loud-
speaker is able to output sounds.

Please write me a letter, if the program is of value for you.
Write also, if you worry about it or if you have further
ideas. You can find my address at the beginning of this descrip-
tion.

Have a lot of fun while testing the program!