Chapter 11 Advanced Batch Files

LEARNING OBJECTIVES

- 1. List the commands used in batch files.
- 2. List and explain batch file rules.
- 3. Explore the function of the REM, PAUSE, and ECHO commands.
- 4. Explain the use of batch files with short cuts.
- 5. Explain the purpose and function of the GOTO command.
- 6. Explain the purpose and function of the SHIFT command.
- 7. Explain the purpose and function of the IF command.
- 8. Explain the purpose and function of the IF EXIST/IF NOT EXIST command.
- 9. Explain the purpose and function of the IF ERRORLEVEL command.
- 10. Explain the purpose and function of writing programs.
- 11. Explain the purpose and function of the environment and environmental variables.
- 12. Explain the use of the SET command.
- 13. Explain the purpose and function of the FOR...IN...DO command.
- 14. Explain the purpose and function of the CALL command.

STUDENT OUTCOMES

- 1. Use the ECHO command to place a blank line in a batch file.
- 2. Use the GOTO command in conjunction with a label to create a loop.
- 3. Use a batch file with a shortcut.
- 4. Use the SHIFT command to move parameters.
- 5. Use the IF command with strings for conditional processing.
- 6. Test for null values in a batch file.
- 7. Use the IF EXIST/IF NOT EXIST command to test for the existence of a file or a subdirectory.
- 8. Use the SET command.
- 9. Use the environment and environmental variables in batch files.
- 10. Use the IF ERRORLEVEL command with XCOPY to write a batch file for testing exit codes.
- 11. Use the FOR...IN...DO command for repetitive processing.
- 12. Use the CALL command in a batch file.

CHAPTER SUMMARY

- 1. You may substitute a double colon (::) for the REM statement.
- 2. To place a blank line in a batch file, use the ECHO command followed immediately by a period (ECHO.).

3. The GOTO command used in conjunction with a label creates a loop. The GOTO will process the command following the label.

- 4. The label in a batch file is not a command, but identifies a location in the batch file.
- 5. The SHIFT command shifts over command line parameters to the left one position at a time.
- 6. The SHIFT command is typically used in conjunction with GOTO and a label.
- 7. The IF command will test for some logical condition. If the condition is true, the command will be processed. If the condition is false, the batch file will fall through to the next line of the batch file.
- 8. The IF command can test whether or not two character strings are identical.
- 9. The IF command can test whether or not a file exists.
- 10. The IF command checks ERRORLEVEL.
- 11. You may also use IF with NOT. The IF NOT command will test for a NOT condition. If a condition is not true, then the command will process. If the command is true, then the batch file will fall through to the next line in the batch file.
- 12. You can test for a null value by using quotation marks, a word, or backslashes.
- 13. To use IF EXIST and to test for the existence of a subdirectory, you may use IF %1\NUL.
- 14. Many programs set an exit code when finished executing. The IF ERRORLEVEL in a batch file will test if an exit code is equal to or greater than the one in the test.
- 15. When you use IF ERRORLEVEL in a batch file, the codes must be listed in descending order.
- 16. When you use IF NOT ERRORLEVEL in a batch file, the codes must be listed in ascending order.
- 17. You may write programs with DEBUG to set exit codes.
- 18. You can write programs directly in DEBUG, or you may write a script file that supplies input for DEBUG to create a program.
- 19. The environment is an area in memory where the operating system leaves messages to itself, including the path and the prompt values.
- 20. You can create environmental variables that can be used in batch files. When using the variable in a batch file, the syntax is *%variable%*.
- 21. The SET command lets you view what is currently in the environment.
- 22. Environmental variables set in other batch files or at the command line remain in effect only during the current Command Prompt session. When the Command Prompt window is closed, the variable disappears.
- 23. The DIRCMD command can be used to preset the DIR command parameters and switches.
- 24. The FOR...IN...DO command allows repetitive processing. The command will execute some command for every value in a specified set.
- 25. When you use FOR...IN...DO at the command line, the syntax is FOR *%variable* IN (*set*) DO *command* [*command-parameter*].
- 26. The /R when used with FOR...IN...DO allows recursive processing. Recursive means that the command will search all the directories.
- 27. Using the /F parameter allows you to select specified text using delimiters that you can set.
- 28. The CALL command allows you to call one batch file from another. When the second batch file is finished executing, it returns control to the first batch file.

KEY TERMS

debug	environmental variable	recursive
EOF (end-of-file) mark	expression	scan code
exit code	loop	script file
conditional processing	null value	variable
environment		

LECTURE NOTES

CHAPTER OUTLINE

Chapter Overview

- Quick review of batch file commands learned in earlier chapters.
- Advanced features of these commands will be explained and used.
- Will explain the purpose and function of remaining batch file commands and then will use these commands to write sophisticated batch files.
- Will refine techniques in working with environment.

BATCH FILE COMMANDS

- Batch File Commands
 - Batch file rules.
 - Has .BAT or .CMD as file extension.
 - Must be ASCII file.
 - Must include legitimate commands.
 - Create generic batch files using replaceable parameters.
 - Are not case-sensitive.
 - Can use any command in batch file that can be used on the command line.
 - Many special batch file commands.
 - See PowerPoint slides # for list of commands and their purposes.
 - Batch files have a limited vocabulary, syntax, and programming logic.
 - Limited in kind of programming they can do.
 - Not as versatile as "real" programming languages.

A REVIEW OF THE REM, PAUSE, AND ECHO COMMANDS

- A Review of the REM, PAUSE, and ECHO Commands
 - REM.
 - Used to **document** batch files.
 - Up to string of 123 characters.
 - ECHO on displays but does not execute what follows REM.
 - Won't display if ECHO is off.
 - Placing REM in front of a command will disable but not delete that specific line.
 - Batch file or CONFIG.SYS file will continue to execute.
 - Disables line without deleting it.
 - PAUSE.

- <Ctrl> + C or <Ctrl> + <Break> interrupts program.
 - Temporarily stops executing batch file.
 - Will not continue until user presses a key.
- Will not do any **conditional processing**.

• ECHO.

- Used on command line or in batch file.
- Controls printing of messages on screen when batch file is run.
- ECHO ON
 - (Default) displays all commands to screen along with output.
 - Useful when tracking operation of a batch file
 - Clutters screen when batch file runs successfully
- ECHO OFF
 - Displays only output of commands to screen.
- ECHO <text string> displays text string to screen.
- Precede ECHO OFF with @ and "ECHO OFF" will not appear on screen.

ADVANCED FEATURES OF ECHO AND REM

- Advanced Features of ECHO and REM
 - REM slows processing of a batch file.
 - Recognized by OS as command and must be processed.
 - Replace REM with double colon (::) for faster processing.
 - Label single colon followed by anything.
 - Using double colon (::) because labels skipped by OS.
 - With ECHO OFF, messages still come on screen.
 - Redirecting output to NUL device eliminates standard output messages.
 - Will not suppress messages like "File not found".
 - To generate blank line on screen:
 - Use ECHO followed by a period (ECHO.).
 - No space between ECHO and period.
 - Using <Enter> will not work in batch files.

ACTIVITY—USING ECHO AND NUL

- Using ECHO and NUL
 - DATA disk in Drive A, A:\> displayed.
 - Activity steps.
 - Use editor to create/save batch file called ONE.BAT press enter only where indicated
 - :: This is a test of a batch file using <Enter>
 - :: different features. <Enter>

COPY CAROLYN.FIL BOOK.FIL <Enter>

<Enter>

TYPE BOOK.FIL <Enter>

ECHO <Enter>

DEL BOOK.FIL <Enter>

COPY NO.FIL BOOK.FIL <Enter>

- Close Editor and key in TYPE ONE.BAT
- Key in: ONE
- Edit/save **ONE.BAT** so it looks as follows:

NUL

@ECHO OFF :: This is a test of a batch file using :: different features COPY CAROLYN.FIL BOOK.FIL > ECHO. TYPE BOOK.FIL ECHO. **DEL BOOK.FIL** COPY NO.FIL BOOK.FIL > NUL

- Key in: ONE
- Activity completed.

THE GOTO COMMAND

- The GOTO Command
 - Use GOTO command to have batch file constructed to behave like a program.
 - Allows **looping** & branching within batch files.
 - Can stop loop
 - Use IF statement or
 - Break into batch file $\langle Ctrl \rangle + C$.
 - Works in conjunction with a **label** to create a **loop**.
 - Processes command following label.
 - Label.
 - Do not confuse with volume label on a disk.
 - Name chosen to flag or identify the location in batch file.
 - Preceded by colon (:).
 - Labels must be unique.
 - Maximum length 8 characters.
 - Not a command.
 - Batch file goes to label carries out command followed on line after label.
 - Not case sensitive but make cases same.
 - Has one parameter GOTO label.
 - Double colon ensures that OS will disregard line since colon may not be used as label name.

ACTIVITY—USING THE GOTO COMMAND

- Using the GOTO Command
 - DATA disk in Drive A with A:\> displayed.
 - Activity steps.
 - Use text editor to create/save batch file called **REPEAT.BAT** so it looks as follows:

REM This file displays many times the contents

REM of a file

:REPEAT

TYPE %1

PAUSE

GOTO REPEAT

- At system prompt, key in: **REPEAT ASTRO.TXT**
- Press **<Enter>** several times to loop through file.
- Press < Ctrl> + C then Y then < Enter>
- Activity completed.

THE SHIFT COMMAND

- The SHIFT Command
 - Number of parameters on command line limited to 10 (%0 %9).
 - 0% represents batch file name itself.
 - Really limited to 9 parameters.
 - SHIFT moves parameters one position to the left, decreasing their number by one.
 - With SHIFT can have unlimited # of parameters.

ACTIVITY—USING THE SHIFT COMMAND

- Using the SHIFT Command
 - DATA disk in Drive A with A:\> displayed.
 - Activity steps.
 - Key in: ECHO a b c d e
 - Use editor to create/save batch file called **ALPHA.BAT** so it looks as follows:

@ ECHO OFF

ECHO %0 %1 %2 %3

SHIFT

ECHO %0 %1 %2 %3

SHIFT

ECHO %0 %1 %2 %3

SHIFT

ECHO %0 %1 %2 %3

- At system prompt key in: **ALPHA** a b c d e f
- Create /save new batch file **UPDATE.BAT** so it looks as follows:

:DOIT

COPY %1/B +> NUL

SHIFT

PAUSE

GOTO DOIT

- Key in: **DIR APR.99**
- Key in: DIR APR.BUD
- Key in: **UPDATE APR.99 APR.BUD**
- Press **<ENTER>** twice then **<Ctrl>** + **C** then **Y**
- Key in: **DIR APR.99**
- Key in: **DIR APR.BUD**
- Create/save a batch file called **SIZE.BAT** so it looks as follows:

:TOP

DIR %1 | FIND "Directory" >> TEMP.FIL

DIR %1 | FIND "bytes" | FIND /V "free" >> TEMP.FIL

SHIFT

GOTO TOP

TYPE TEMP.FIL

PAUSE

DEL TEMP.FIL

- Key in: SIZE CLASS TRIP
- Press <Ctrl> + C then Y

- Key in: **TYPE TEMP.FIL** | **MORE**
- Press $\langle Ctrl \rangle + C$
- Activity completed.

THE IF COMMAND

- The IF Command
 - Allows conditional processing of parts of batch files.
 - Compares two items determines if they are identical, or if one is greater than the other.
 - Comparison yields one of two values.
 - True items are identical
 - False items are not identical.
 - Syntax: **IF** <**condition**> <**command**>.
 - Condition true Command executed.
 - Condition false:
 - Command not executed.
 - Batch file falls through to next command line in batch file.
 - IF can check for three conditions.
 - Whether two sets of characters are or are not identical.
 - Whether or not a file exists.
 - Value of variable in ERRORLEVEL
 - ERRORLEVEL is a number that a program can set depending on the outcome of a process.

THE IF COMMAND USING STRINGS

- The IF Command using Strings
 - IF can be used to compare strings.
 - Two equal signs (= =) separate what is to be compared.
 - Tell IF statement to GOTO a label or perform an operation whether the condition is true or false.

ACTIVITY—USING THE IF COMMAND WITH STRINGS

- Using the IF Command with Strings
 - DATA disk in Drive A with A:\> displayed.
 - Activity steps.
 - Use editor to create/save batch file called **GREET.BAT** so it looks as follows: (No spaces between

two equal signs.)

IF %1==Carolyn GOTO Carolyn

IF %1==Bette GOTO Bette

ECHO Isn't anyone there?

GOTO FINISH

:Carolyn

ECHO Greetings, Ms. Carolyn.

GOTO FINISH

:Bette

ECHO Greetings, Ms. Bette.

:FINISH

• Key in:

- GREET Carolyn then GREET Bette
- GREET JUAN then GREET BETTE
- Activity completed.

TESTING FOR NULL VALUES

- Testing for NULL Values
 - Activity 11.11 tested for exact match of character strings.
 - If used SHIFT until all parameters used end up in endless loop.
 - Can test to see if string matches.
 - Can test for a **null value** = literally testing for nothing (no data).
 - Even though nothing is there, there must be "something" to confirm the "nothing."
 - To test for "nothing" must place a value in test that will give you nothing.
 - Variety of methods for testing null values.
 - Use quotation marks so that your statement becomes:
 - IF "%1"= ="" GOTO LABEL
 - Above means, "If nothing there, GOTO somewhere else."
 - Use void: (can use any word)
 - IF %1void= = GOTO LABEL
 - Use backslash (\):
 - •IF \%1\= =\\ GOTO LABEL

ACTIVITY—USING NULL VALUES

- Using NULL Values
 - *Note:* DATA disk in Drive A with A: \> displayed.
 - Activity steps.
 - Edit and save file called **UPDATE.BAT** to look as follows:

:DOIT

IF "%1"=="" GOTO END

COPY %1/B +> NUL

SHIFT

PAUSE

GOTO DOIT

:END

- Key in:
 - DIR CAROLYN.FIL
 - UPDATE CAROLYN.FIL
- Press **<Enter>**
- Key in: **DIR CAROLYN.FIL**
- Edit/save **SIZE.BAT** file to look as follows:

:TOP

IF %1nothing==nothing GOTO END

DIR %1 | FIND "Directory" >> TEMP.FIL

DIR %1 | FIND "bytes" | FIND /V "free" >> TEMP.FIL

SHIFT

GOTO TOP

TYPE TEMP.FIL

PAUSE

DEL TEMP.FIL

:END

• Key in:

- DEL TEMP.FIL
- SIZE CLASS TRIP
- TYPE TEMP.FIL
- Activity completed.

THE IF EXIST/IF NOT EXIST COMMAND

- The IF EXIST/IF NOT EXIST Command
 - Checks for existence/non-existence of a specified file.
 - Using IF EXIST command.
 - If file exists
 - Condition is true
 - Processing passes to specified GOTO location or to command following IF statement.
 - If file does not exist:
 - Condition is false.
 - OS ignores command in IF clause.
 - Batch process reads next line in file.
 - Using IF NOT EXIST command.
 - If file does not exist:
 - Condition is true.
 - Processing passes to specified GOTO location or to command following IF NOT statement.
 - If file exists:
 - Condition is false.
 - Batch process reads next line in file.

ACTIVITY—USING IF EXIST TO TEST FOR A FILE

- Using IF EXIST to Test for a File
 - *Note:* DATA disk in Drive A with A: \> displayed.
 - Activity steps.
 - Use any text editor to create/save a file called RENDIR.BAT to look as follows:

IF \%1\==\\ **GOTO** end

IF NOT \%2\==\\ GOTO next

ECHO You must include a destination name

ECHO for the new directory name.

GOTO end

:next

IF EXIST %1 GOTO message

REN %1 %2

GOTO end

:message

ECHO This is a file, not a directory.

:end

- Key in: RENDIR JAN.99 LAST
- Key in: **RENDIR TEST OLDER**
- Edit RENDIR.BAT file so IF NOT EXIST%1\NUL GOTO message replaces IF EXIST %1GOTO message.
- Key in: **RENDIR TEST OLDER**

Edit/save **SIZE.BAT** file to look as follows:

IF EXIST TEMP.FIL DEL TEMP.FIL

:TOP

IF %1nothing==nothing GOTO END

IF NOT EXIST %1\NUL GOTO NEXT

DIR %1 | FIND "Directory" >> TEMP.FIL

DIR %1 | FIND "bytes" | FIND /V "free" >> TEMP.FIL

:NEXT

SHIFT

GOTO TOP

:END

TYPE TEMP.FIL

PAUSE

DEL TEMP.FIL

- Key in: SIZE CLASS JUP.PAR TRIP
- Press <Enter>
- Activity completed.

THE IF ERRORLEVEL COMMAND TESTING

- The IF ERRORLEVEL Command Testing
 - Program can set exit code when it finishes executing.
 - Batch file tests this exit code with IF ERRORLEVEL statement.
 - Exit code doesn't test for a match with ERRORLEVEL, but determines if it is greater than or equal to it.
 - Test IF ERRORLEVEL 0 will always be true every possible exit code is greater than or equal to 0.

ACTIVITY—USING IF ERRORLEVEL WITH COPY

- Using IF ERRORLEVEL with COPY
 - DATA disk in Drive A with A:\> displayed.
 - Activity steps.
 - Use any text editor to create/save a file called **ERROR.BAT** to look as follows:

COPY %1 %2

IF ERRORLEVEL 1 GOTO NOTOK

IF ERRORLEVEL 0 GOTO OK

:NOTOK

ECHO There are no %1 files. Try again.

GOTOEND

:OK

ECHO You copied the %1 files

successfully.

:END

- Key in: ERROR *.TXT OLDER
- Key in: ERROR *.NON OLDER
- Activity completed.

WRITING PROGRAMS TO TEST FOR KEY CODES

- Writing Programs to Test for Key Codes
 - Exit codes.
 - Set by OS.
 - Created by writing small programs that are based upon some activity.
 - Can write program that identifies which key on a keyboard was pressed and report which

key it was.

- Can do this activity because each time key is pressed- identified by a 1- or 2-digit scan code.
- Two things reported when you press a key on the keyboard:
 - First, that a key was pressed.
 - Second, that the key was released.
- Going to write a program that will report the scan code for any key that is pressed.
 - Once know reported code, can test for a specific key by using ERRORLEVEL in the batch file.
 - Batch file then acts based upon the reported code.
 - Program must be written in 0's and 1's the "bits and bytes" the computer understands.
- Several ways to write a program.
 - Use programming language.
 - Use an OS utility program called DEBUG.
- DEBUG.
 - Allows testing and debugging of executable files those with .COM or .EXE file extensions.
 - Small program that has its own commands and syntax you could write a .COM program directly with DEBUG.
- Unless a student is a programming expert, they probably don't want to do this.
- Easiest way to use DEBUG is to create a script file.
 - Script = set of instructions that you can write with any ASCII editor.
 - Once the script is written, you can "feed" it to the DEBUG program via redirection (**DEBUG < SCRIPT.FIL**).
 - DEBUG will then convert the script file to an executable program with a .COM file extension.
 - This process is the easiest way to create a file that will report the scan code for any key that is pressed.
 - The program that will be created will be called REPLY.COM (Activity 11.19).

ACTIVITY—WRITING A SCRIPT FILE

- Writing a Script File
 - DATA disk in Drive A with A:\> displayed.
 - Activity steps.
 - Use any text editor to create/save a file called **REPLY.SCR** to look as follows:

e 100 b4 08 cd 21 3c 00 75 02 cd 21 b4 4c cd

21

rcx

 \mathbf{e}

n reply.com

 \mathbf{w}

q

- Key in: **DEBUG < REPLY.SCR**
- Key in: **DIR REPLY.COM**
- Use any text editor to create/save a file called **KEYING.BAT** to look as follows:

ECHO PRESS F1 TO CLEAR THE SCREEN.

ECHO PRESS F2 TO DISPLAY THE DIRECTORY.

ECHO PRESS ANY OTHER KEY TO EXIT.

REPLY

IF ERRORLEVEL 61 GOTO END
IF ERRORLEVEL 60 GOTO F2
IF ERRORLEVEL 59 GOTO F1
GOTO END
:F1
CLS
GOTO END
:F2
DIR
:END
• Key in:

- **KEYING** then **<F1>** then **KEYING** then **<F2>**
- Activity completed.

THE ENVIRONMENT

- The Environment
 - Area OS sets aside in memory where data can be stored.
 - Acts as a scratch pad where notes kept about items OS system needs to know.
 - OS places/keeps location of file COMMAND.COM in environment.
 - Applications can read items in environment and can post own messages there.
 - Programs can get value of variable and use it to modify their operation.
 - OS can store data in memory.
 - Two strings:
 - Name of variable.
 - Value of variable.
 - Environmental variable is name assigned to string (value) of data.
 - User can set environmental variables.
 - Some common variables set when Windows started.
 - USERNAME, USERPROFILE, PATH, PROMPT.
 - Location of CMD.EXE file.
 - User can also leave messages in the environment.
 - User leaves a message by using the SET command.
 - Environmental variables set in OS remain in effect for entire work session.
 - Environmental variables set in Command Prompt window or in batch files executed in the MS-DOS window will remain in effect **only** during that command prompt session.
 - While values are in effect, use the syntax %variable%.
 - SET syntax: SET [variable=[string]]
 - Variable specifies environment -variable name.
 - String specifies series of characters to assign to the variable.
 - SET without parameters displays current environment variables.
 - SET command followed by a letter lists any environmental variables that begin with that letter.

ACTIVITY—USING SET AND THE ENVIRONMENTAL VARIABLES

- Using SET and the Environmental Variables
 - If environment display is too long to fit on one screen use MORE filter.
 - Activity steps.
 - Key in: **SET**

- If used MORE filter return to prompt or press Q
- Key in:
 - SET U then SET S
 - ECHO %PATH%
 - ECHO %SystemRoot%
 - C: then CD %systemroot%
 - CD %userprofile% then DIR/AH
 - DIR "local settings"\TEMP\~*.tmp
 - DEL %TEMP%\~*.tmp
 - DIR %TEMP%\~*.tmp
 - **CD** \ then **A**:
- Activity completed.

USING SET AND THE ENVIRONMENT IN BATCH FILES

- Using SET and the Environment in Batch Files
 - Can use built-in environmental variables that Windows sets/uses.
 - Can set own environmental variables.
 - Can give them name/value in batch file or command line.
 - Set variables only good for that session of Command Prompt Window.

ACTIVITY—USING SET AND THE ENVIRONMENT IN BATCH FILES

- Using SET and the Environment in Batch Files
 - DATA disk in Drive A with A:\> displayed.
 - Activity steps.
 - Close Command Prompt window
 - Reopen Command Prompt window and return to A prompt.
 - Create/save **TESTIT.BAT** batch file that looks as follows:

@ECHO OFF

ECHO %PATH%

ECHO.

- Key in:
 - TESTIT
 - SET TODAY=C:\WUGXP*.FP
 - SET
- Create/save **SETTING.BAT** batch file that looks as follows:

DIR %todav%

ECHO %TODAY%

- Key in:
 - SETTING
 - SET today=C:\WUGXP\.TMP
 - **SETTING** then **SET TODAY**=
 - SET T then SET TODAY=MONDAY
 - ECHO %TODAY% then SET T
- Close Command Prompt window
- Reopen Command Prompt window and return to A prompt.
- Key in: SET T
- Create/save **ADD.BAT** batch file that looks as follows:

IF "%1"=="" GOTO END

PATH > OLDPATH.BAT

:TOP

PATH %PATH%;%1

SHIFT

IF NOT \%1\==\\ GOTO TOP :END

- Key in:
 - PATH > ORIGPATH.BAT
 - ADD A:\ then ORIGPATH
 - ADD A:\;A:\OLDER;A:\CLASS
 - PATH then ORIGPATH
- Activity completed.

THE DIRCMD ENVIRONMENTAL VARIABLE

- The DIRCMD Environmental Variable
 - Can place/use variables in environment.
 - DIRCMD environmental variable.
 - With SET to preset DIR command parameters/switches.
 - With ERRORLEVEL to write batch files to change way DIR displays information for current MS-DOS work session.

ACTIVITY—USING DIRCMD

- Using DIRCMD
 - DATA disk in Drive A with A:\> displayed.
 - Activity steps.
 - Create/save batch file MY.BAT

@ECHO OFF

CLS

ECHO.

ECHO.

ECHO How do you want your directory displayed?

ECHO.

ECHO 1. Files only arranged by file name.

ECHO 2. Files only arranged by file name.

ECHO 3. Files only arranged by file extension.

ECHO 4. Files only arranged by file extension.

Z to A

ECHO 5. Directory displays in default mode.

ECHO.

ECHO PLEASE SELECT A NUMBER.

ECHO. REPLY

ECHO.

IF ERRORLEVEL 49 IF NOT ERRORLEVEL
IF ERRORLEVEL 50 IF NOT ERRORLEVEL
IF ERRORLEVEL 51 IF NOT ERRORLEVEL
IF ERRORLEVEL 52 IF NOT ERRORLEVEL
IF ERRORLEVEL 53 IF NOT ERRORLEVEL 54
SET DIRCMD=/O- E/A-II

- Key in:
 - MY then 4 then SET D
 - DIR CLASS then MY then 5
 - SET D
- Activity completed.

THE FOR...IN...DO COMMAND

- The FOR...IN...DO Command
 - Can be issued at command line or placed in batch file.
 - Allows repetitive processing within batch files.
 - FOR allows you to use a single command to issue several commands at once.
 - The command can DO something FOR every value in a specified set.
 - FOR...IN...DO syntax (at command line):

FOR %variable% IN (set) DO command

[command-parameters]

- % variable specifies a replaceable parameter.
- (set) specifies a set of one or more files. Wildcards may be used.
- Command specifies the command to carry out for each file.
- Command-parameters specifies parameters or switches for the specified command.
- FOR...IN...DO syntax (batch program):
 - Use %%variable instead of %variable.
 - Batch file variable arbitrary single letter.
 - Double percent sign with letter (%%a) distinguishes batch file variable from replaceable variable (%1).
- Allows the user to loop through a file for predefined list of values.
- OS gets values from position in command line.
- The set enclosed in parentheses.
 - Values in set used to DO some command.
 - Items in set separated by spaces or commas.
 - Wildcards may be used.

ACTIVITY—USING THE FOR...IN...DO COMMAND

- Activity: Using FOR...IN...DO Command.
 - DATA disk in Drive A and A:\> displayed.
 - Note 2: Look at command line going to use in Step 1. In English, command says: Using the variable %a to hold each value in the set (what is in the parentheses), do the command (Type) to each value in the set (%a).
 - Activity steps.
 - Key in: FOR %a IN (*.99) DO TYPE %a
 - Key in: FOR %x IN (APR99 NOFILE.EXT D.BAT) DO TYPE %x
 - Key in: FOR %y IN (APR.99,NOFILE.EXT,D.BAT) DO TYPE %y
 - Key in: TYPE *.99 NOFILE.EXT
 - Key in: FOR %a IN (*.99,NOFILE) DO IF EXIST %a TYPE %a
 - Create/save batch file **DO.BAT** that looks as follows:

FOR %%v IN (Patty Nicki Sandy Brian) DO ECHO %%v

- Key in: **DO**
- Create/save a batch file called **PASS.BAT** that looks as follows:

FOR %%a IN (%USERS%) DO IF "%1"=="%%a" GOTO OKAY

:NO

ECHO You, %1, are NOT allowed in the system.

GOTO END

:OKAY

ECHO Welcome %1 to my world of computers.

:END

- Key in: SET USERS=Carolyn, Bette
- Key in: PASS Bette
- Key in: PASS Denzel
- Edit/save **UPDATE.BAT** to look as follows:

:DOIT

IF "%1"=="" GOTO END

FOR %%v IN (%1) DO COPY %%v/b + > NUL

SHIFT

PAUSE

GOTO DOIT

:END

- Key in: MD BATCH
- Key in: **ADD A:\BATCH**
- Key in: MOVE *.BAT BATCH
- Key in: **MOVE REPLY*.* BATCH**
- Key in: DIR *.SWT
- Key in: DIR *.CAP
- Key in: UPDATE *.SWT *.CAP
- Press **<Enter>** until returned to prompt
- Key in: **DIR *.SWT *.CAP**
- Activity completed.

MORE FEATURES OF THE FOR...IN...DO COMMAND

- More Features of the FOR...IN...DO Command
 - May list environmental variables so they are divided and appear on separate lines.
 - Use /R parameter (recursive).
 - Command will search and perform actions on all subdirectories beneath it.
 - Use tilde operator (~) to:
 - Strip a file name of quotation marks.
 - Expand a variable.
 - Select specific text from ASCII files.

ACTIVITY—USING THE ADDITIONAL FEATURES OF THE FOR...IN...DO **COMMAND**

- Using Additional Features of the FOR...IN...DO Command
 - *Note:* DATA disk in Drive A with A: \> displayed.
 - Activity steps.
 - Key in:
 - SET PATHEXT
 - FOR %a IN (%pathext%) DO @ ECHO %a
 - FOR %a IN ("Sandy and Patty.txt", MERCUYRY.TXT) DO
 - @ECHO %a FOR %a IN ("Sandy and Patty.txt", MERCURY.TXT) DO @ECHO %~a
 - FOR %a IN ("Sandy and Patty.txt", %a "CZG %a" MERCURY.TXT) DO @REN
 - FOR %a IN ("Sandy and Patty.txt", MERCURY.TXT) DO @ECHO %~a
 - DIR CZG*.*
 - DEL CZG*.*
 - Key in following and then press <Enter>: FOR %a IN ("Sandy and Patty.txt", JUPITER.TXT) DO REN %a "CZG %~a"
 - Key in:
 - COPY C:\UGXP*.TXT TRIP
 - FOR /R %a IN (SANDY*.*) DO @ECHO %a

FOR /R %a IN (SANDY*.*) DO @ECHO %~nxa

• In batch file directory create PREFIX.BAT batch file. Use own name instead of "YourNameHere"

@ECHO OFF

REM YourNameHere

REM Purpose of batch file is to add a new prefix to any file name.

IF"1"==""GOTOMESSAGE

IF"2"==""GOTOMESSAGE2

FOR /R %%a IN (%2) DO REN "%%~a" "%1 %%~nxa"

GOTOEND

:MESSAGE

ECHO You must include a prefix you wish to use.

ECHO Syntax is PREFIX prefix filename

GOTOEND

:MESSAGE2

ECHO You must include a file name you wish to rename.

ECHO Syntax is PREFIX prefix filename

GOTOEND

:MESSAGE2

ECHO You must include a file name you wish to rename.

ECHO Syntax is PREFIX prefix filename

:END

- Be sure BATCH directory is in your path. Can add **ADD.BAT** to include it.
- In root of A:\ key in: DIR Sandy*.* /S
- Key in:
 - PREFIX RBP SANDY*
 - DIR *SANDY* /S then DIR RBP* /S
- Create **PERSON.BAT** batch file in **BATCH** directory that looks as follows:

@ECHO OFF

FOR /F "TOKENS=1,2,7" %% a IN (%1) DO

ECHO %%b %%a %%c

- Key in:
 - CD \ then TYPE PERSONAL.FIL
 - PERSON PERSONAL.FIL
- Edit **PERSON.BAT** file as follows:

@ECHO OFF

SORT < %1 > %2

FOR /F "TOKENS=1,2,7" %%a IN (%2) DO

ECHO %%b %%a, %%c

DEL %2

- Key in: PERSON PERSONAL.FIL TEMP.FIL
- Edit PERSONAL.FIL and select the first five lines and copy them into a new file called SHORT.FIL
- Exit Edit and key in: PERSON SHORT.FIL TEMP.FIL
- Edit **SHORT.FIL** file as follows:

Gillay; Carolyn; 699 Lemon; Orange; CA; Professor

Panezich;Frank;689 Lake;Orange;CA;Teacher

Tuttle; Steven; 356

Embassy; Mission

Viejo;CA;Juggler

Maurdeff; Kathryn; 550

Traver; Ann Arbor; MI; Teacher

Maurdeff;Sonia;550

Traver; Ann Arbor; MI; Student

• Close SHORT.FIL Edit PERSON.BAT to look as follows:

@ECHO OFF

SORT < %1 > %2

FOR /F "usebackq delilms=;TOKENS=1,2,6" %%a IN (%2) DO ECHO %%b %%a,

%%c

DEL %2

- Key in: PERSON SHORT.FIL TEMP.FIL
- Activity completed.

THE CALL COMMAND

- The CALL Command
 - CALL allows you to run one batch file from within another.
 - Second batch file is finished executing returns control to first batch file.
 - Without CALL, the original batch file will not continue after the second one is run.

ACTIVITY—USING CALL

- Using CALL
 - DATA disk in Drive A with A:\> displayed.
 - You have executed command A:\BATCH\>ADD A:\BATCH at some time during current Command Prompt session.
 - Activity steps.
 - Key in:
 - CD BATCH
 - COPY CON BELL.BAT
 - **ECHO** <Ctrl> + **G** <Enter> 6 times
 - <**F6**>
 - TYPE BELL.BAT
 - RELL
 - Create/save a batch file called **BELLING.BAT** in **BATCH** directory that looks as follows:

COPY *.ABC *.XYZ

DEL *.XYZ

BELL

- Key in: **BELLING**
- Edit/save **BELLING.BAT** that looks as follows:

COPY *.ABC *.XYZ

BELL

REM You are about to delete the *.XYZ files. Are you sure?

PAUSE

DEL *.XYZ

- Key in: **BELLING** then **DIR** *.XYZ
- Edit/save **BELLING.BAT** that looks as follows:

COPY *.ABC *.XYZ

CALL BELL

REM You are about to delete the *.XYZ files. Are you sure?

PAUSE

DEL *.XYZ

- Key in: BELLING
- Press **<Enter>**
- Key in: **DIR** *.XYZ
- Use COPY CON to create/save a batch file called HOME.DAT that looks as follows:

COPY CON HOME.DAT

SET HOME=<F6>

• Create/save **HOMETO.BAT** batch file in batch directory that looks as follows:

COPY A:\BATCH\HOME.DAT

A:\BATCH\HOMESAVE.BAT

CD >> A:\BATCH\HOMESAVE.BAT

CALL HOMESAVE.BAT

DEL A:\BATCH\HOMESAVE.BAT

- Create/save a batch file **HOME.BAT** that contains the following: **CD %HOME%**
- Key in:
 - PATH then SET H then CD \
 - CD WORK\ADS then HOMETO
 - SET H then CD \MEDIA\TV
 - HOME then CD \ then ORIGPATH
- Activity completed.

Activity 11.6 - Using the GOTO Command

- Line 1 REM This file displays many times the contents
- Line 2 **REM of a file**
- Line 3 -: REPEAT
- Line 4 **TYPE %1**
- Line 5 PAUSE
- Line 6 -GOTO REPEAT

First two lines are remarks that will not execute.

Not including ECHO OFF because you want to see what is happening in batch file.

Omitting ECHOOFF is way to "debut a batch program

Third line: REPEAT) is a label – must be preceded by a colon

Fourth line simple TYPE Command with replaceable parameter

Fifth line - Pause command is placed here so you may see what is happening.

Sixth and last line is the loop.

The GOTO tells batch file to return to label. (:REPEAT)

It will then return to line 4 and execute the TYPE Com and

It will then read lines 5 and 6 and continually repeat the process.

Activity 11.15: Using IF EXIST to Test for a FILE

Batch File by Line Number	Test TRUE	Processing	Test FALSE
1. IF \% 1\= =\\ GOTO end	User keys in nothing for %1. Since test is true, action is to go to line 12	Testing for null value.	User keys in value for %1. Since test is false, action is to go to line 2.
2. IF NOT $\%2=\GOTO$ next	User keys in nothing for %2. Since test is true, action is to go to line 3.	Testing for null value	User keys in value for %2. Since test is false, action is to go to line 6.
3. ECHO You must include a destination name		Message for user that he	

or she did not include a value.

4. ECHO for the new directory name.

Continuation of the message.

5. GOTO end

Falls through to the GOTO end statement. Action is to go to line 12.

6. :next

Label referred to in line 2.

7. IF EXIST %1 GOTO message

User keys in file name for %1. Since test is true, action is to go to line 10.

Testing for value for %1. Is it a file or a directory?

User keys in directory for %1. Since test is false, action is to go to line 8.

8. REN %1 %2

Since %1 test is false (not a file), renaming directory can proceed.

9. GOTO end

After directory is renamed, falls through to GOTO end.

10. :message

Label referred to in line 2.

11. ECHO This is a file, not a

directory.

Message that user used a file name, not a directory name.

12. :end