|  |  |
| --- | --- |
| **Navigation:** [*TCC*](https://jpsoft.com/help/tcc.htm) **>** [*Commands*](https://jpsoft.com/help/commands.htm)**FOR** | **Scroll** [Prev](https://jpsoft.com/help/font.htm) [Top](https://jpsoft.com/help/commands.htm) [Next](https://jpsoft.com/help/free.htm) **More**  |
| ***Purpose:*** | Repeat a command for several values of a variable |

|  |  |
| --- | --- |
| ***Format:*** | File and string mode |

FOR [range...] [/I"text"] [/A:[[-|+]rhsadecijopt /D /F ["options"] /H /Nj /O:[-]adegnrstu /R [path]  [/T"delimiters"] /W ] %var IN ([@]set) DO command | (command ... [LEAVEFOR] )

Counted mode

FOR /L %var IN (start, step, end) DO command | (command ... [LEAVEFOR] )

|  |  |
| --- | --- |
| ***options*** | Parsing options for a "file parsing" FOR. |

|  |  |
| --- | --- |
| ***range*** | One or more [range](https://jpsoft.com/help/ranges.htm) specifications |

|  |  |
| --- | --- |
| ***path***  | The starting directory for a "recursive" FOR. |

|  |  |
| --- | --- |
| ***%var***  | The variable to be used in the command ("FOR variable"). |

|  |  |
| --- | --- |
| ***set***  | A set of values for the variable. |

|  |  |
| --- | --- |
| ***start***  | The starting value for a "counted" FOR. |

|  |  |
| --- | --- |
| ***step***  | The increment value for a "counted" FOR. |

|  |  |
| --- | --- |
| ***end***  | The limit value for a "counted" FOR. |

|  |  |
| --- | --- |
| ***command*** | A command or group of commands to be executed for each value of the variable. |

|  |  |
| --- | --- |
| [/A:  (Attribute select)](https://jpsoft.com/help/for.htm#a) | [/N](https://jpsoft.com/help/for.htm#o) (defaults) |
| [/D(irectories only)](https://jpsoft.com/help/for.htm#d) | [/O:... (Order)](https://jpsoft.com/help/for.htm#o) |
| [/F(ile parsing)](https://jpsoft.com/help/for.htm#f) | [/R(ecursive)](https://jpsoft.com/help/for.htm#r) |
| [/H(ide dots)](https://jpsoft.com/help/for.htm#h) | [/T (delimiter list)](https://jpsoft.com/help/for.htm#t) |
| [/I description range](https://jpsoft.com/help/for.htm#i) | [/W(ildcards)](https://jpsoft.com/help/for.htm#w) |
| [/L (counted loop)](https://jpsoft.com/help/for.htm#l) |   |

***File Selection***

Supports [attribute switches](https://jpsoft.com/help/attrswitch.htm), extended [wildcards](https://jpsoft.com/help/wildcards.htm), [ranges](https://jpsoft.com/help/ranges.htm), [multiple file names](https://jpsoft.com/help/multfiles.htm), and [include lists](https://jpsoft.com/help/incllist.htm).

Ranges must appear immediately after the FOR keyword after alias expansions (if any), and only affect the selection of files specified using wildcards.

Use wildcards with caution on LFN volumes; see [LFN File Searches](https://jpsoft.com/help/lfnsearch.htm) for details.

***Usage:***

FOR begins by creating a ***set***. It then executes a command for every member of ***set***. The command can be an internal command, an alias, an external command, or a batch file. The members of ***set*** can be a list of file names, text strings, a group of numeric values, or text read from a list of files.

When ***set*** is made up of text or several separate file names (not an include list), the elements must be separated by spaces, tabs, or commas.

FOR includes a large number of options, some of which duplicate functions available in other internal commands. It also supports additional conventions not found in our other commands, included for compatibility with CMD.

The first three sections below ([Working with Files](https://jpsoft.com/help/for.htm#for_files), [Working with Text](https://jpsoft.com/help/for.htm#for_text), and [Retrieving Text from Files](https://jpsoft.com/help/for.htm#for_retrieve)) describe the FOR command and the enhancements to it which are included in ***TCC***. The sections on [Parsing Text from Files](https://jpsoft.com/help/for.htm#for_parse) and [Counted FOR Loops](https://jpsoft.com/help/for.htm#for_count) describe features added for compatibility with CMD. The sections [Directory Recursion](https://jpsoft.com/help/for.htm#dirrecurs) and [Output Redirection](https://jpsoft.com/help/for.htm#or) warn of special considerations. The section entitled [Other Notes](https://jpsoft.com/help/for.htm#for_other) contains information you may need if you use any aspect of the FOR command extensively.

FOR sets two internal variables:

|  |  |
| --- | --- |
| %\_for\_files | The number of files processed |

|  |  |
| --- | --- |
| %\_for\_errors | The number of errors |

If the [Duplicate CMD Bugs](https://jpsoft.com/help/inistartupdlg.htm) configuration option is set, ***TCC*** will emulate undocumented CMD behavior when FOR ***set*** arguments are split across multiple lines. For example:

for %a in (

one

two

three

) do (

echo %a

)

**Working with Files**

Normally, ***set*** is a list of files specified with wildcards. For example, if you use this line in a batch file:

for %x in (\*.txt) list %x

Then [LIST](https://jpsoft.com/help/list.htm) will be executed once for each file in the current directory with the extension ***.****TXT*. The FOR variable %x is set equal to each of the file names in turn, then the LIST command is executed for each file. (You could do the same thing more easily with a simple LIST \*.TXT*.* We used FOR here so you could get a feel for how it operates, using a simple example. Many of the examples in this section are constructed in the same way.)

***Set*** can include multiple files and include lists, like this:

for %x in (d:\\*.txt;\*.doc;\*.asc e:\test\\*.txt;\*.doc) type %x

FOR supports [wildcards and extended wildcards](https://jpsoft.com/help/wildcards.htm), as well as [extended parent directory](https://jpsoft.com/help/pardir.htm) names, *e.g.*, *...\\*.txt* to process all of the ***.****TXT* files that are contained in the directory 2 levels above the current directory.

By default those members of ***set*** that include wildcards match only files, not directories.

When you use FOR on an LFN drive, you must quote any file names within set which contain white space or special characters. The same restriction may apply to names returned in the FOR variable, if you pass them to ***TCC*** internal commands, or other commands which require quoting filenames with white space. FOR does not quote returned names automatically, even if you included quotes in set.

If set includes filenames, the file list can be further refined by using date, time, size, description and file exclusion [ranges](https://jpsoft.com/help/ranges.htm). The range or ranges must be placed immediately after the word FOR. Ranges affect only those members of set which contain wildcards. For example, the FOR below will process all of the *\*.TXT* files that were created or updated on December 4, 2010, and of the file ABC.LST  regardless of its timestamp:

for /[d12-4-2010,+0] %x in (\*.txt abc.lst) ...

If ***command*** is an internal command that supports ranges, an independent range can also be used in ***command*** itself.

You can also refine the list by limiting it with the [/A:](https://jpsoft.com/help/for.htm#a) option to select only files that have specific attributes.

When you use wildcards to specify ***set***, FOR scans the directory and finds each file which matches the wildcard name(s) you specified. If, during the processing of the FOR command, you create a new file that could be included in ***set***, it may or may not appear in a some later iteration of the same FOR command. Whether or not the new file appears depends on its physical location in the directory structure. For example, if you use FOR to execute a command for all .*TXT* files, and the command also creates one or more new .*TXT* files, those new files may or may not be processed during the current FOR command, depending on where they are placed in the physical structure of the directory. This is a Windows constraint over which ***TCC*** has no control. Therefore, in order to achieve consistent results you should construct FOR commands which do not create files that could become part of set for the current command.

**Working with Text**

***Set*** can also be made up of text instead of file names. For example, to create three files named *file1*, *file2*, and *file3*, each containing a blank line:

for %suffix in (1 2 3) echo. > file%suffix

You can also use the names of environment variables as the text. This example displays the name and content of several variables from the environment (see the general discussion of the [Environment](https://jpsoft.com/help/environment.htm) for details on the use of square brackets when expanding environment variables):

for %var in (path prompt comspec) echo %var=%[%var]

**Retrieving Text from Files**

If the name of a file in ***set*** is prefixed with **@** ("at" sign), it is considered as an [@file list](https://jpsoft.com/help/atfile.htm)***.*** FOR extracts each line from the file and places it in the FOR variable.

**Warning*:*** if the line contains characters which are syntactically significant for ***TCC***, for example, one of the characters <"[]|>, it may have undesirable effects. You may use the /X option of [SETDOS](https://jpsoft.com/help/setdos.htm) to mitigate them.

If you use **@CON** as the filename, FOR will read from standard input (typically a redirected input file) or from a pipe. If you use **@CLIP:** as the filename, FOR will read any text available from the Windows clipboard. See [Redirection and Piping](https://jpsoft.com/help/redirpipe.htm) for more information on these features.

See [@file list](https://jpsoft.com/help/atfile.htm) for additional details.

**Parsing Text from Files**

Another method of working with text from files is to have FOR parse each line of each file for you. To begin a file-parsing FOR, you must use the [/F](https://jpsoft.com/help/for.htm#f) option and include one or more file names in set. When you use this form of FOR, the variable name must be a single letter, for example, **%a**.

This method of parsing, included for compatibility with CMD, can be cumbersome and inflexible. For a more powerful method, use FOR with [@filename](https://jpsoft.com/help/f_filename.htm) as the ***set*** to retrieve each line from the file, as described in the previous section, and use variable functions like [@FIELD](https://jpsoft.com/help/f_field.htm), [@INSTR](https://jpsoft.com/help/f_instr.htm), [@LEFT](https://jpsoft.com/help/f_left.htm), [@RIGHT](https://jpsoft.com/help/f_right.htm), and [@WORD](https://jpsoft.com/help/f_word.htm) to parse the line (see [Variable Functions](https://jpsoft.com/help/functions.htm) for information on variable functions).

By default, FOR will extract the first word or token from each line and return it in the variable. For example, to display the first word on each line in the file *FLIST.TXT*:

for /f %a in (flist.txt) echo %a

You can control the way FOR /F parses each line by specifying one or more parsing options in a quoted string immediately after the **/F**. The available options are:

**skip=*n***:  FOR /F will skip ***n*** lines at the beginning of each file before parsing the remainder of the file.

**tokens=*n*, *m*, ...:** By default, FOR /F returns just the first word or ***token*** from each parsed line in the variable you named. You can have it return more than one token in the variable, or return tokens in several variables, with this option.

This option is followed by a list of numbers separated by commas. The first number tells FOR /F which token to return in the first variable, the second number tells it which to return in the second variable, etc. The variables follow each other alphabetically starting with the variable you name on the FOR command line. This example returns the first word of each line in TEST.TXT in ***%d***, the second in ***%e***, and the third in ***%f***:

for /f "tokens=1,2,3" %d in (test.txt) ...

You can also indicate a range of tokens by separating the numbers with a hyphen **-**.

**eol=*c***:If FOR /F finds the character ***c*** in the line, it will assume that the character and any text following it are part of a comment and ignore the rest of the line.

**delims=*xxx..***:By default, FOR /F sees spaces, tabs and commas as word or token delimiters. This option replaces those delimiters with all of the characters following the equal sign to the end of the string. This option must therefore be the last one used in the quoted options string.

**usebackq** : Duplicates the awkward CMD syntax. A back quoted string is executed as a command; a single quoted string is a literal string; and double quotes quote filenames in the file set. We don't recommend **usebackq** for batch files written for ***TCC***, as ***TCC*** has much more elegant ways of doing the same things.

You can also use FOR /F to parse a single string instead of each line of a file by using the string, in quotes, as ***set.*** For example, this command will assign variable ***A*** to the string ***this***, ***B*** to ***is***, etc., then display ***this***:

for /f "tokens=1,2,3,4" %a in ("this is a test") echo %a

**"Counted" FOR Loop**

The "counted FOR" loop is included for compatibility with CMD. In most cases, you will find the [DO](https://jpsoft.com/help/do.htm) command more useful for performing counted loops.

In a counted FOR command, the ***set*** is made up of numeric values instead of text or file names. To begin a counted FOR command, you must use the [/L](https://jpsoft.com/help/for.htm#l) option and then include three values, separated by commas, in ***set***. These are the ***start***, ***step***, and ***end*** values. During the first iteration of the FOR loop, the variable is set equal to the ***start*** value. Before each iteration, the variable is increased by the ***step*** value. The loop ends when the variable exceeds the ***end*** value. This example will print the numbers from 1 to 10:

for /l %val in (1,1,10) echo %val

This example will print the odd numbers from **1** to **10** (**1**, **3**, **5**, **7**, and **9**):

for /l %val in (1,2,10) echo %val

The ***step*** value can be negative. If it is, the loop will end when the variable is less than the ***end*** value.

Numeric input may be entered in either decimal format (a sequence of 0-9 digits) or in hexadecimal format ("0x" followed by a sequence of 0-F hex digits).

***WARNING!*** You must not have white space between ***start*** and the subsequent comma, nor between ***step*** and its subsequent comma. White space after the comma is accepted.

**Directory Recursion**

By default, FOR works only with files in the current directory or a specified directory. Option switch [/R](https://jpsoft.com/help/for.htm#r) specifies that the search should  recursively process subdirectories. If you specify a directory name immediately after [/R](https://jpsoft.com/help/for.htm#r), FOR will start in that directory and then search each of its subdirectories. If no directory is specified after the /R, the search starts in the current default directory. If you do specify a directory, *and* its name includes any special characters, it must be enclosed in double quotes. For example, it must be quoted if it is specified with the aid of an environment variable, e.g., ***%windir\command***.

There are two differences in the invocation of ***command*** caused by directory recursion:

•The loop control variable contains the full name of the matching file

•***command*** is executed with the default directory set to the directory in which the file was found

This example processes all ***.****TXT* files in the current directory and its subdirectories:

for /r %x in (\*.txt) ...

This example works with all of the ***.****BAK* files on drive ***D:***

for /r d:\ %x in (\*.bak) ...

**Output Redirection**

The default output redirection (i.e., **for ... > filename**) creates a new output file in each iteration. If ***filename*** does not include an absolute file path, it will be created relative to the then current default directory. If you use directory recursion, this path will change for each directory processed. The simplest way to force a single target file is to enclose the whole command in parentheses, e.g.,:

(for %x in (set) command) > filename

**Other Notes**

•You can use either **%** or **%%** in front of the variable name (***var***) in the command. Either form will work, whether the FOR command is typed from the command line or is part of an alias or batch file. (CMD which requires a single **%** if FOR is used at the command line, but requires %% if FOR is used in a batch file.) Note that you must have at least one **%** sign present.

•The variable name can be up to 80 characters long.

•If the FOR command is an alias, e.g., **alias for=\*for /h**, [range](https://jpsoft.com/help/ranges.htm) specifications will be ignored.

•The word DO is unnecessary but accepted. Do not confuse it with the completely unrelated [DO](https://jpsoft.com/help/do.htm) command.

•If the name of the FOR variable ***var*** is a single character, for compatibility with CMD, it is created in the environment in a special way that does not overwrite an existing environment variable with the same name. Wherever ***command*** contains the % sign immediately followed by the character which is the name of the FOR variable, it is replaced by its value, regardless of any characters following it. For example, the following command tries to add ***a:*** and ***b:*** to the end of [PATH](https://jpsoft.com/help/path_var.htm), but will not work as intended:

**for %p in (a: b:) path %path;%p**

**path**

b:ath;b:

The ***%p*** in ***%path*** was interpreted as the FOR variable ***%p*** followed by the text ***ath***, not what was intended. To get around this, use a different letter or a longer name for the FOR variable, or use square brackets around the variable name, as shown in the examples below, any one of which accomplishes the original goal:

for %p in (a: b:) path %[path];%p

for %x in (a: b:) path %path;%x

for %px in (a: b:) path %path;%px

•If the name of the FOR variable contains more than one character, it is created in the environment, and erased when FOR is completed, whether or not a variable by that name existed before the FOR. It  cannot be modified with the [SET](https://jpsoft.com/help/set.htm), [ESET](https://jpsoft.com/help/eset.htm), or [UNSET](https://jpsoft.com/help/unset.htm) commands. If you already had a variable with that name, it will no longer be accessible. For example, a command that begins

for %path in ...

will write over your current [PATH](https://jpsoft.com/help/path_var.htm) setting, then erase the [PATH](https://jpsoft.com/help/path_var.htm) variable completely when FOR is done.

•***Command*** may also use the FOR variable with the special syntax of CMD described in [Special syntax for CMD compatibility](https://jpsoft.com/help/cmdsyntax.htm).

•The following example uses FOR with variable functions to delete the *.BAK* files for which a corresponding *.TXT* file exists in the current directory (this should be entered on one line):

for %file in (\*.txt) del %@name[%file].bak

The above command may not work properly on an LFN drive, because the returned *FILE* variable might contain white space. To correct this problem, you need two sets of quotes, one for [DEL](https://jpsoft.com/help/del.htm) and one for [%@NAME](https://jpsoft.com/help/f_name.htm):

for %file in (\*.txt) del "%@name["%file"].bak"

•You can use [command grouping](https://jpsoft.com/help/grouping.htm) to execute multiple commands for each element in ***set***. For example, the following command copies each *.WKQ* file in the current directory to the *D:\WKSAVE* directory, then changes the extension of each file in the current directory to ***.SAV***:

[for %file in (\*.wkq) (copy %file d:\wksave\ & ren %file \*.sav)

or (in a batch file):

for %file in (\*.wkq) (

 copy %file d:\wksave\

 ren %file \*.sav

)

•In a batch file you can use [GOSUB](https://jpsoft.com/help/gosub.htm) to execute a subroutine for every element in ***set***. Within the subroutine, the FOR variable can be used just like environment variable. This is a convenient way to execute a complex sequence of commands for every element in ***set*** without [CALL](https://jpsoft.com/help/call.htm)ing another batch file.

•One unusual use of FOR is to execute a collection of batch files or other commands with the same parameter. For example, you might want to have three batch files all operate on the same data file. The FOR command could look like this:

for %cmd in (filetest fileform fileprnt) %cmd datafile

This line will expand to three separate commands:

filetest datafile

fileform datafile

fileprnt datafile

•FOR statements can be nested.

**LEAVEFOR**

The special keyword LEAVEFOR can be used inside a FOR command group. LEAVEFOR terminates the current FOR processing and continues with the line following the FOR command, in a manner similar to that of the LEAVE keyword in a [DO](https://jpsoft.com/help/do.htm) command.

for %i in (\*) (

 ...

 if "%i" == "xyz.abc" leavefor

...

)

***Options:***

|  |  |
| --- | --- |
| **/A:** | Process only those files that have the specified attribute(s). **/A:** will be used only when processing wildcard file names in ***set***. It will be ignored for filenames without wildcards or other items in ***set***. See [Attribute Switches](https://jpsoft.com/help/attrswitch.htm) for information on the attributes which can follow **/A:**. |

For example, to process only those files with the archive attribute set:

for /a:a %f in (\*) echo %f needs a backup!

Default: **/A:-D-H-S**, i.e. include only ***files*** without the ***hidden*** and ***system*** attributes.

You can specify **/A:=** to display a dialog to help you set individual attributes.

|  |  |
| --- | --- |
| **/D** | Only return subdirectories, excluding "**.**" and "**..**" . |

|  |  |
| --- | --- |
| **/F** | Return one or more words or tokens from each line of each file in ***set***. The **/F** option can be followed by one or more options in a quoted string which control how the parsing is performed. See [Parsing Text From Files](https://jpsoft.com/help/for.htm#for_parse). |

|  |  |
| --- | --- |
| **/H** | Suppresses the assignment of the "." and ".." directories to the FOR variable when directories are explicitly includedusing the [/A:](https://jpsoft.com/help/for.htm#a) option.  |

|  |  |
| --- | --- |
| **/I"*text*"** | Select filenames by matching text in their descriptions. See [Description Ranges](https://jpsoft.com/help/descriptionranges.htm). |

|  |  |
| --- | --- |
| **/L** | Interpret the three values in ***set*** as the ***start***, ***step***, and ***end*** values of a counted loop. See [Counted FOR Loops](https://jpsoft.com/help/for.htm#for_count). |

|  |  |
| --- | --- |
| **/Nj** | Don't recurse into symlinks or junctions (see [/R](https://jpsoft.com/help/for.htm#r)). |

|  |  |
| --- | --- |
| **/O:...** | Sort the files before processing. |

You may use any combination of the sorting options below. If multiple options are used, the listing will be sorted with the first sort option as the primary key, the next as the secondary key, and so on:

|  |  |
| --- | --- |
| **n** | Sort by filename and extension, unless **e** is explicitly included. This is the default. |

|  |  |
| --- | --- |
| **-** | Reverse the sort order for the next sort key |

|  |  |
| --- | --- |
| **a** | Sort names and extensions in standard ASCII order, instead of numerically when numeric substrings are included in the name or extension. |

|  |  |
| --- | --- |
| **d** | Sort by date and time (oldest first); also see **/T:acw** |

|  |  |
| --- | --- |
| **e** | Sort by extension |

|  |  |
| --- | --- |
| **g** | Group subdirectories first, then files |

|  |  |
| --- | --- |
| **r** | Reverse the sort order for all options |

|  |  |
| --- | --- |
| **s** | Sort by size |

|  |  |
| --- | --- |
| **t** | Same as **d** |

|  |  |
| --- | --- |
| **u** | Unsorted |

The /O:... option saves all of the matching filenames and then performs the requested operation. This avoids the potential problem of processing files more than once.

|  |  |
| --- | --- |
| **/R [*path*]** | Look in the current directory and all of its subdirectories for files in ***set***. If the **/R** is followed by a directory name, look for files in that directory and all of its subdirectories. **Warning:** if the directory name includes special characters, including "%" to indicate an environment variable, it must be enclosed in double quotes ("). /R supports Windows shell folder names; see [CDD](https://jpsoft.com/help/cdd.htm) for details. |

|  |  |
| --- | --- |
| **/T"*text*"** | Specify the delimiters to be used when parsing a string set. |

|  |  |
| --- | --- |
| **/W** | The FOR set is to be processed as filenames, even if no wildcards are detected. (This is useful if you want to use regular expressions with FOR.) |



|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |    |  |