Piping

You can create a “pipe” to send the standard output of one command to the standard input of another command:

`command1 | command2Send`	the standard output of command1 to the standard input of command2
`command1 |& command2Send`	the standard output and standard error of command1 to the standard input of command2

For example, to take the output of the **SET** command (which displays a list of your environment variables and their values) and pipe it to the **SORT** utility to generate a sorted list, you would use the command:

```
[c:\] set | sort
```

To do the same thing and then pipe the sorted list to the internal **LIST** command for full-screen viewing:

```
[c:\] set | sort | list
```

The **TEE** and **Y** commands are “pipe fittings” which add more flexibility to pipes.

Like redirection, pipes are fully nestable. For example, you can invoke a batch file and send all of its output to another command with a pipe. A pipe on a command within the batch file will take effect for that command only; when the command is completed, output will revert to the pipe in use for the batch file as a whole. You may also have 2 or more pipes operating simultaneously if, for example, you have the pipes running in different windows.

CMD.EXE implements pipes by starting a new process for the receiving program instead of using temporary files. The sending and receiving programs run simultaneously; the sending program writes to the pipe and the receiving program reads from the pipe. When the receiving program finishes reading and processing the piped data, it is ends automatically.

When you use pipes with **CMD.EXE** make sure you think about any possible consequences that can occur from using a separate process to run the receiving program.