Introduction to Windows Commands(part-1)

Table of contents:

Windows Commands

File and Directory Commands:

1. **dir - List files and directories in the current directory.**
2. **cd - Change the current directory.**
3. **mkdir - Create a new directory.**
4. **rmdir - Remove a directory.**

5. copy - Copy files or directories.
6. move - Move files or directories.
7. del - Delete files.
8. ren - Rename files or directories.
9. type - Display the contents of a text file.
10. find - Search for a specific string in a file.
11. attrib - Change file attributes.
12. tree - Display directory structure as a tree.
13. xcopy - Extended copy command with more options.
14. chkdsk - Check and repair disk errors.
15. fc - Compare two files or sets of files.
16. comp - Compare the contents of two files.
17. robocopy - Robust file and directory copying tool.
18. sfc - System File Checker to repair corrupted system files.
19. findstr - Search for specific strings in files.
20. more - Display the contents of a text file one page at a time.
21. sort - Sort the contents of a text file.
22. xcopy /e - Copy directories and subdirectories, including empty ones.
23. compact - Compress or decompress files on an NTFS partition.
24. xcaccls - Backup and restore NTFS permissions.
25. subst - Associate a drive letter with a directory.
26. deltree - Delete a directory and its subdirectories.
27. cipher - Display or alter file encryption on NTFS volumes.
28. fsutil - File system utility for managing various file system settings.

29. openfiles - Display or disconnect open shared files and folders.

Network Commands:

30. ipconfig - Display network configuration information.
31. ping - Test network connectivity.
32. tracert - Trace the route to a remote host.
33. netstat - Display network statistics.
34. nslookup - Look up IP addresses and domain names.
35. hostname - Display or set the computer's hostname.
36. arp - Display and modify the ARP cache.
37. route - Display or modify the routing table.
38. telnet - Connect to remote hosts using Telnet.
39. ftp - Transfer files to/from remote FTP servers.
40. net - Manage network resources.
41. netsh - Network Shell for configuring network-related settings.
42. net use - Connect or disconnect a computer from shared resources.
43. net view - Display a list of available network resources.
44. net share - Create, delete, or manage shared folders.
45. net session - View and manage network sessions.
46. net time - Synchronize the computer's time with a network server.
47. netdom - Domain-related management tool.
48. route print - Display the routing table with more details.
49. nbtstat - Display statistics and current connections using NetBIOS over TCP/IP.
50. ipconfig /flushdns - Flush and reset the DNS resolver cache.
51. ipconfig /release - Release the current DHCP configuration.

52. ipconfig /renew - Renew the DHCP configuration.
53. netsh firewall - Configure the Windows Firewall.
54. netstat -a - Display all active network connections and listening ports.

System Information and Management:

55. systeminfo - Display detailed system information.
56. tasklist - List running processes.
57. taskkill - Terminate processes or applications.
58. msconfig - System Configuration Utility.
59. regedit - Registry Editor.
60. eventvwr - Event Viewer.
61. services.msc - Services management console.
62. shutdown - Shut down or restart the computer.
63. gpupdate - Update Group Policy settings.
64. ver - Display the Windows version.
65. systeminfo - Display detailed system information.
66. gpresult - Display Group Policy settings for the current user.
67. powercfg - Configure power management settings.
68. bcdedit - Boot Configuration Data Editor for managing boot options.
69. dxdiag - DirectX Diagnostic Tool for troubleshooting DirectX issues.
70. driverquery - List installed device drivers.
71. msinfo32 - System Information utility.
72. mmc - Microsoft Management Console for creating custom management tools.
73. taskmgr - Task Manager for managing running processes.
74. perfmon - Performance Monitor for system monitoring.

75. wmic - Windows Management Instrumentation Command-line tool.
76. schtasks - Schedule tasks to run at specific times or events.

User Account Management:

77. net user - Manage user accounts.
78. net group - Manage user groups.
79. net localgroup - Manage local groups.
80. whoami - Display the current user.
81. runas - Run a program as another user.
82. control userpasswords2 - User Accounts control panel.

Disk and Storage Management:

83. diskpart - Disk Partitioning tool.
84. format - Format a disk drive.
85. defrag - Defragment disk drives.
86. diskmgmt.msc - Disk Management console.
87. cleanmgr - Disk Cleanup utility.

Miscellaneous Commands:

88. cls - Clear the screen.
89. echo - Display text on the screen.
90. date - Display or set the system date.
91. time - Display or set the system time.
92. help - Get help on commands.
93. color - Change the console text and background color.
94. assoc - Display or modify file extension associations.

95. shutdown - Shutdown or restart the computer.
96. shutdown /s - Shutdown the computer (immediate).
97. shutdown /r - Restart the computer (immediate).
98. shutdown /h - Hibernate the computer (if supported).

1. dir - List files and directories in the current directory.

1.1. List Files in Current Directory

- Command: `dir`

1.2. List Files with Detailed Information

- Command: `dir /q`

1.3. List Files with Hidden and System Files

- Command: `dir /a`

1.4. List Files in Long Format

- Command: `dir /w`

---

2. Listing Specific File Types

2.1. List All `.txt` Files

- Command: `dir *.txt`

2.2. List All Files with Specific Extension (e.g., `.jpg`)

- Command: `dir *.jpg`

2.3. List Files in Subdirectories

- **Command:** `dir /s`

---

3. Sorting Directory Listings

3.1. Sort by File Size

- **Command:** `dir /o:s`

3.2. Sort by Date and Time

- **Command:** `dir /o:d`

3.3. Sort in Reverse Order

- **Command:** `dir /o:-d`

---

4. Displaying File Attributes

4.1. Display File Attributes with File Names

- **Command:** `dir /a`

4.2. Display Only Directories

- **Command:** `dir /ad`

4.3. Display Only Hidden Files

- **Command:** `dir /ah`

---

5. Advanced File Listing

5.1. List Files with Summary

- **Command:** `dir /b`

5.2. List Files with Directory Count

- **Command:** `dir /s /b`

5.3. Display File Names Only

- **Command:** `dir /b /a-d`

---

6. Directory Navigation

6.1. List Files in Parent Directory

- **Command:** `dir ..`

6.2. List Files in a Specific Directory

- **Command:** `dir C:\Windows`

---

7. File Searching

7.1. Search for a File in Current Directory

- **Command:** `dir filename*`

7.2. Search for Files with Wildcards

- **Command:** `dir *.*`

7.3. Search for Files in Subdirectories

- **Command:** `dir /s filename`

---

8. Redirecting Output

8.1. Redirect Output to a Text File

- **Command:** `dir > output.txt`

8.2. Append Output to an Existing File

- **Command:** `dir >> output.txt`

---

9. Recursive Directory Listing

9.1. List Files and Folders in Subdirectories Recursively

- **Command:** `dir /s /b`

9.2. List Files in All Subdirectories Including Hidden

- **Command:** `dir /s /ah`

---

10. List Files with Specific Time

10.1. List Files Modified After a Specific Date

- **Command:** `dir /t:w`

10.2. List Files Based on Last Accessed Time

- **Command:** `dir /t:a`

---

11. Pagination and Formatting

11.1. Show Directory Listing One Page at a Time

- **Command:** `dir /p`

11.2. Display Results in Columns

- **Command:** `dir /w`

---

12. Displaying File Ownership

12.1. List Files with Owner Information

- **Command:** `dir /q`

12.2. List Files in Long Format with Ownership

- **Command:** `dir /q /s`

---

13. Filtering Files by Size

13.1. Display Files Larger than a Specific Size

- **Command:** `dir /s /o:-s /b`

13.2. Display Files Smaller than a Specific Size

- **Command:** `dir /s /o:+s /b`

---

14. Displaying Time and Date

14.1. Display Directory Files and Folders with Date and Time

- **Command:** `dir /t:c`

14.2. Display Directory Files Sorted by Date Modified

- **Command:** `dir /o:d`

---

15. File Grouping and Counting

15.1. Display Directory Grouped by File Types

- **Command:** `dir /o:g`

15.2. Count Total Number of Files in a Directory

- **Command:** `dir /s /a-d | find /c /i "file"`

---

16. Special File Listings

16.1. Display Only Executable Files

- **Command:** `dir *.exe`

16.2. Display Only Directories

- **Command:** `dir /ad`

---

17. Displaying Long Listing with File Sizes

17.1. List Files with Detailed File Sizes

- **Command:** `dir /s /b /o:s`

17.2. Display File Sizes in Bytes

- **Command:** `dir /s /o:s /q`

---

18. Directory Size Calculation

18.1. List Files and Calculate Total Size

- **Command:** `dir /s /a`

18.2. Calculate Size of Directory and Files

- **Command:** `dir /s`

---

19. Listing Files with Permissions

19.1. List Files with Detailed Permissions

- **Command:** `dir /q`

19.2. List Files in Directory with Group Permissions

- **Command:** `dir /q /a`

---

20. Excluding Files from Listing

20.1. Exclude Files Matching Specific Pattern

- **Command:** `dir /a-d | findstr /v "*.exe"`

20.2. Exclude Directories from Listing

- **Command:** `dir /a /b | findstr /v /c:"\."`

---

2. cd - Change the current directory.

1.1 Change to a Specific Directory

- **Command:** `cd [path]`
- **Description: Changes the current working directory to the specified path. For example,** `cd C:\Users\YourName\Documents` **will change the directory to** `Documents`**.**

1.2 Change to the Root Directory

- **Command:** `cd \`
- **Description: Changes the current directory to the root directory of the current drive. For example, on the** `C:` **drive, it will navigate to** `C:\`**.**

---

2. Directory Navigation with Drive Letters

2.1 Change to a Directory on Another Drive

- **Command:** `cd D:\FolderName`
- **Description: Navigates to a folder on a different drive (e.g., from** `C:` **to** `D:`**). You must specify the drive letter along with the path.**

2.2 Change to a Directory Using a Relative Path

- **Command:** `cd FolderName`
- **Description: Changes the current directory to a subfolder of the current directory. For example,** `cd Documents` **will move to the** `Documents` **subfolder in the current directory.**

3.1 Navigate Up One Directory Level

- Command: `cd ..`
- Description: Moves up one directory level. For example, if you are in `C:\Users\YourName\Documents`, running `cd ..` will take you to `C:\Users\YourName`.

3.2 Navigate to the Previous Directory

- Command: `cd -`
- Description: Navigates to the last visited directory. This command works similarly to using a back button in a file explorer.

---

4. Viewing Current Directory

4.1 Display the Current Directory Path

- Command: `cd`
- Description: Displays the full path of the current working directory.

---

5. Error Handling and Special Cases

5.1 Attempt to Change to a Nonexistent Directory

- Command: `cd [nonexistentDirectory]`
- Description: If you try to change to a directory that doesn't exist, you'll get an error message stating, "The system cannot find the path specified."

5.2 Change Directory with Spaces in Path

- Command: `cd "C:\Program Files\My Application"`
- Description: If the directory path contains spaces, enclose the path in double quotes to avoid errors.

---

6. Switching Between Drives

6.1 Switch to Another Drive

- Command: `D:`

- Description: To change from `C:` to another drive like `D:`, simply type the drive letter followed by a colon. This will change the current drive to `D:`.

3. mkdir - Create a new directory.

1.1 Create a Single Directory

- Command: `mkdir MyFolder`
- Description: Creates a new directory named `MyFolder` in the current working directory.

Example:
bash
Copy code
```
mkdir MyFolder
```

- Result: A folder named `MyFolder` will be created in the current directory.

1.2 Create a Directory with a Path

- Command: `mkdir C:\Users\YourName\Documents\NewFolder`
- Description: Creates a directory `NewFolder` at the specified path.

Example:
bash
Copy code
```
mkdir C:\Users\YourName\Documents\NewFolder
```

- Result: The folder `NewFolder` will be created inside `Documents`.

---

2. Creating Multiple Directories

2.1 Create Multiple Directories at Once

- Command: `mkdir Folder1 Folder2 Folder3`
- Description: Creates `Folder1`, `Folder2`, and `Folder3` in the current working directory.

Example:
bash

Copy code
```
mkdir Folder1 Folder2 Folder3
```

- Result: Three directories (`Folder1`, `Folder2`, and `Folder3`) will be created in the current directory.

2.2 Create Directories with Nested Structure

- Command: `mkdir ParentFolder\ChildFolder1 ParentFolder\ChildFolder2`
- Description: Creates a parent folder with child folders inside it.

Example:
bash
Copy code
```
mkdir ParentFolder\ChildFolder1 ParentFolder\ChildFolder2
```

- Result: A folder `ParentFolder` will be created with two child folders `ChildFolder1` and `ChildFolder2`.

---

3. Directory Creation with Spaces in Names

3.1 Create a Directory with Spaces in the Name

- Command: `mkdir "My New Folder"`
- Description: Creates a folder with spaces in its name by enclosing the name in double quotes.

Example:
bash
Copy code
```
mkdir "My New Folder"
```

- Result: A folder named `My New Folder` will be created in the current directory.

3.2 Create Multiple Directories with Spaces

- Command: `mkdir "Folder 1" "Folder 2" "Folder 3"`
- Description: Creates multiple directories with spaces in their names. The names should be enclosed in double quotes.

Example:
bash

Copy code
```
mkdir "Folder 1" "Folder 2" "Folder 3"
```

- Result: Three directories (`Folder 1`, `Folder 2`, and `Folder 3`) will be created in the current directory.

---

4. Error Handling and Special Cases

4.1 Attempt to Create an Existing Directory

- Command: `mkdir ExistingFolder`
- Description: If the directory already exists, an error message will be displayed.

Example:
bash
Copy code
```
mkdir ExistingFolder
```

- Error: `"The directory already exists."`

4.2 Invalid Characters in Directory Name

- Command: `mkdir My*Folder`
- Description: Throws an error if the directory name contains invalid characters, like `*`, `?`, `|`, etc.

Example:
bash
Copy code
```
mkdir My*Folder
```

- Error: `"The directory name is invalid."`

---

5. Creating Parent Directories (with `/p` Option)

5.1 Create Parent Directories if They Do Not Exist

- Command: `mkdir C:\NewFolder\SubFolder\ChildFolder`
- Description: Creates parent directories if they do not exist. This is useful when creating a deeply nested folder structure.

Example:
bash
Copy code
```
mkdir C:\NewFolder\SubFolder\ChildFolder
```

- **Result: If `C:\NewFolder` and `C:\NewFolder\SubFolder` do not exist, they will be created along with `ChildFolder`.**

---

6. Viewing Created Directories

6.1 List Created Directories

- **Command: `dir`**
- **Description: After creating a directory with `mkdir`, use the `dir` command to list all directories and files in the current location to verify the creation.**

Example:
bash
Copy code
```
dir
```

- **Result: Lists the directories, including the newly created ones, in the current directory.**

---

7. Creating Hidden Directories

7.1 Create a Hidden Directory

- **Command: `mkdir C:\Users\YourName\Documents\.HiddenFolder`**
- **Description: While Windows doesn't treat directories starting with a period (`.`) as hidden by default, this is useful for Unix-like systems or if you want to manually hide the folder.**

Example:
bash
Copy code
```
mkdir C:\Users\YourName\Documents\.HiddenFolder
```

- **Result: A hidden directory `.HiddenFolder` will be created inside `Documents`.**

4. rmdir - Remove a directory.

1.1 Remove an Empty Directory

- **Command:** `rmdir [DirectoryName]`
- **Description: Removes an empty directory. If the directory contains files or subdirectories, it will not be removed.**

Example:
bash
Copy code
```
rmdir MyFolder
```

- **Result: The folder `MyFolder` will be removed, but only if it's empty.**

1.2 Remove a Directory with a Full Path

- **Command:** `rmdir C:\Users\YourName\Documents\OldFolder`
- **Description: Removes the specified directory at the given path. It must be empty to be removed.**

Example:
bash
Copy code
```
rmdir C:\Users\YourName\Documents\OldFolder
```

- **Result: The empty directory `OldFolder` located in `Documents` will be removed.**

---

2. Removing Non-Empty Directories

2.1 Remove a Directory and Its Contents (with `/s` option)

- **Command:** `rmdir /s [DirectoryName]`
- **Description: Removes a directory and all of its contents, including subdirectories and files. Be cautious, as this action is permanent.**

Example:
bash
Copy code
```
rmdir /s OldFolder
```

- **Result: The `OldFolder` and everything inside it will be removed.**

2.2 Remove a Directory and Its Contents with a Full Path (with `/s` option)

- **Command: `rmdir /s C:\Users\YourName\Documents\OldFolder`**
- **Description: Removes the directory at the specified path and all of its contents.**

Example:
bash
Copy code

```
rmdir /s C:\Users\YourName\Documents\OldFolder
```

- **Result: The folder `OldFolder` and everything within it will be deleted from `Documents`.**

---

3. Forcing Removal

3.1 Force Removal without Confirmation (with `/q` option)

- **Command: `rmdir /s /q [DirectoryName]`**
- **Description: Removes a directory and its contents without asking for confirmation. This can be useful for automating the process in scripts.**

Example:
bash
Copy code

```
rmdir /s /q OldFolder
```

- **Result: The `OldFolder` and its contents will be removed without any prompts or confirmation messages.**

3.2 Force Removal with Full Path

- **Command: `rmdir /s /q C:\Users\YourName\Documents\OldFolder`**
- **Description: Removes the specified directory and its contents without any confirmation prompts.**

Example:
bash
Copy code

```
rmdir /s /q C:\Users\YourName\Documents\OldFolder
```

- **Result: The folder `OldFolder` and all its contents in `Documents` will be removed quietly.**

---

4. Error Handling and Special Cases

4.1 Attempt to Remove a Non-Empty Directory Without `/s` Option

- **Command:** `rmdir MyFolder`
- **Description:** Attempts to remove a directory that is not empty will result in an error.

Example:
bash
Copy code
```
rmdir MyFolder
```

- **Error:** `"The directory is not empty."`

4.2 Attempt to Remove a Directory That Doesn't Exist

- **Command:** `rmdir NonExistentFolder`
- **Description:** An error will occur if the specified directory does not exist.

Example:
bash
Copy code
```
rmdir NonExistentFolder
```

- **Error:** `"The system cannot find the path specified."`

---

5. Viewing Directory After Removal

5.1 List Remaining Directories

- **Command:** `dir`
- **Description:** After removing a directory, use the `dir` command to list the remaining files and directories in the current location.

Example:
bash
Copy code
```
dir
```

- **Result: Displays the contents of the current directory, confirming that the target directory has been removed.**

---

6. Removing Hidden Directories

6.1 Remove a Hidden Directory

- **Command: `rmdir /s /q C:\Users\YourName\Documents\.HiddenFolder`**
- **Description: Hidden directories can be removed in the same way as regular directories by specifying the full path. Ensure that hidden folder attributes are removed if necessary.**

Example:
bash
Copy code
```
rmdir /s /q C:\Users\YourName\Documents\.HiddenFolder
```

- **Result: The hidden directory `.HiddenFolder` will be removed along with all its contents.**

Here is the `copy` command guide that you can copy:

---

5. copy - Copy files or directories.

1.1 Copy a File to Another Location

- **Command: `copy [SourceFile] [Destination]`**
- **Description: Copies a file from the source location to the specified destination.**

Example:
bash
Copy code
```
copy C:\Users\YourName\Documents\File1.txt D:\Backup\
```

- **Result: The file `File1.txt` from the `Documents` folder will be copied to the `Backup` folder on the D: drive.**

1.2 Copy a File with a New Name

- **Command:** `copy [SourceFile] [NewFileName]`
- **Description:** Copies a file and gives it a new name in the destination folder.

Example:
bash
Copy code
```
copy C:\Users\YourName\Documents\File1.txt D:\Backup\File1_Copy.txt
```

- **Result:** The file `File1.txt` will be copied and renamed as `File1_Copy.txt` in the `Backup` folder.

---

2. Copy Multiple Files

2.1 Copy All Files from a Directory

- **Command:** `copy [SourceFolder]\* [Destination]`
- **Description:** Copies all files from the source folder to the specified destination folder.

Example:
bash
Copy code
```
copy C:\Users\YourName\Documents\* D:\Backup\
```

- **Result:** All files from the `Documents` folder will be copied to the `Backup` folder on the D: drive.

2.2 Copy Files with a Specific Extension

- **Command:** `copy [SourceFolder]\*.ext [Destination]`
- **Description:** Copies all files of a specific extension from the source folder to the destination folder.

Example:
bash
Copy code
```
copy C:\Users\YourName\Documents\*.txt D:\Backup\
```

- **Result:** All `.txt` files from the `Documents` folder will be copied to the `Backup` folder on the D: drive.

---

3. Overwriting Files

3.1 Overwrite Existing Files without Confirmation

- Command: `copy /y [SourceFile] [Destination]`
- Description: Copies a file and overwrites any existing file in the destination without asking for confirmation.

Example:
bash
Copy code
```
copy /y C:\Users\YourName\Documents\File1.txt D:\Backup\
```

- Result: The file `File1.txt` from the `Documents` folder will be copied to `Backup`, overwriting any existing file with the same name.

3.2 Prompt Before Overwriting

- Command: `copy /-y [SourceFile] [Destination]`
- Description: Prompts for confirmation before overwriting any existing file in the destination.

Example:
bash
Copy code
```
copy /-y C:\Users\YourName\Documents\File1.txt D:\Backup\
```

- Result: If a file with the same name exists in `Backup`, it will ask for confirmation before overwriting it.

---

4. Copying Files with Hidden or System Attributes

4.1 Copy Hidden Files

- Command: `copy /h [SourceFile] [Destination]`
- Description: Copies hidden files along with other files.

Example:
bash
Copy code
```
copy /h C:\Users\YourName\Documents\HiddenFile.txt D:\Backup\
```

- **Result: The hidden file `HiddenFile.txt` will be copied to the `Backup` folder on the D: drive.**

4.2 Copy System Files

- **Command: `copy /s [SourceFile] [Destination]`**
- **Description: Copies system files along with regular files.**

Example:
bash
Copy code
```
copy /s C:\Users\YourName\Documents\SystemFile.sys D:\Backup\
```

- **Result: The system file `SystemFile.sys` will be copied to the `Backup` folder on the D: drive.**

---

5. Copy Files Across Different Drives

5.1 Copy a File Between Different Drives

- **Command: `copy [SourceFile] [DestinationDrive]:\`**
- **Description: Copies a file from one drive to another.**

Example:
bash
Copy code
```
copy C:\Users\YourName\Documents\File1.txt D:
```

- **Result: The file `File1.txt` from the `Documents` folder will be copied to the root of the `D:` drive.**

---

6. Using Wildcards

6.1 Copy Files Using Wildcards

- **Command: `copy [SourceFolder]\[Wildcard] [Destination]`**
- **Description: Uses wildcards (`*`, `?`) to copy multiple files matching the pattern.**

Example:
bash

Copy code
```
copy C:\Users\YourName\Documents\*.txt D:\Backup\
```

- Result: All `.txt` files from the `Documents` folder will be copied to the `Backup` folder on the D: drive.

6. move - Move files or directories

1.1 Move a File to Another Location

- Command: `move [SourceFile] [Destination]`
- Description: Moves a file from the source location to the specified destination.

Example:
bash
Copy code
```
move C:\Users\YourName\Documents\File1.txt D:\Backup\
```

- Result: The file `File1.txt` from the `Documents` folder will be moved to the `Backup` folder on the D: drive.

1.2 Move a File with a New Name

- Command: `move [SourceFile] [DestinationFile]`
- Description: Moves a file to the destination folder and optionally renames it.

Example:
bash
Copy code
```
move C:\Users\YourName\Documents\File1.txt D:\Backup\File1_Moved.txt
```

- Result: The file `File1.txt` will be moved and renamed as `File1_Moved.txt` in the `Backup` folder.

---

2. Moving Directories

2.1 Move a Directory to Another Location

- Command: `move [SourceDirectory] [Destination]`

- **Description: Moves a directory and all of its contents from the source location to the destination.**

Example:
bash
Copy code
```
move C:\Users\YourName\Documents\OldFolder D:\Backup\
```

- **Result: The entire folder `OldFolder` from the `Documents` folder will be moved to the `Backup` folder on the D: drive.**

2.2 Move a Directory to a New Location with a New Name

- **Command: `move [SourceDirectory] [DestinationFolder]\NewFolderName`**
- **Description: Moves a directory and renames it in the destination folder.**

Example:
bash
Copy code
```
move C:\Users\YourName\Documents\OldFolder D:\Backup\NewFolderName
```

- **Result: The folder `OldFolder` will be moved to the `Backup` folder and renamed as `NewFolderName`.**

---

3. Overwriting Existing Files

3.1 Move and Overwrite Files Without Confirmation

- **Command: `move /y [SourceFile] [Destination]`**
- **Description: Moves a file and overwrites any existing file in the destination without asking for confirmation.**

Example:
bash
Copy code
```
move /y C:\Users\YourName\Documents\File1.txt D:\Backup\
```

- **Result: The file `File1.txt` from the `Documents` folder will be moved to the `Backup` folder on the D: drive, overwriting any existing file with the same name.**

3.2 Prompt Before Overwriting Files

- **Command: `move /-y [SourceFile] [Destination]`**

- **Description: Prompts for confirmation before overwriting any existing file in the destination.**

Example:
bash
Copy code
```
move /-y C:\Users\YourName\Documents\File1.txt D:\Backup\
```

- **Result: If a file with the same name exists in `Backup`, it will ask for confirmation before overwriting it.**

---

4. Using Wildcards

4.1 Move Files Using Wildcards

- **Command: `move [SourceFolder]\[Wildcard] [Destination]`**
- **Description: Moves files matching the wildcard pattern from the source folder to the destination folder.**

Example:
bash
Copy code
```
move C:\Users\YourName\Documents\*.txt D:\Backup\
```

- **Result: All `.txt` files from the `Documents` folder will be moved to the `Backup` folder on the D: drive.**

---

5. Moving Files Across Different Drives

5.1 Move a File Between Different Drives

- **Command: `move [SourceFile] [DestinationDrive]:\`**
- **Description: Moves a file from one drive to another.**

Example:
bash
Copy code
```
move C:\Users\YourName\Documents\File1.txt D:
```

- **Result: The file `File1.txt` from the `Documents` folder will be moved to the root of the D: drive.**

---

6. Moving Files with Hidden or System Attributes

6.1 Move Hidden Files

- **Command:** `move /h [SourceFile] [Destination]`
- **Description:** Moves hidden files along with regular files.

Example:
bash
Copy code
```
move /h C:\Users\YourName\Documents\HiddenFile.txt D:\Backup\
```

- **Result:** The hidden file `HiddenFile.txt` will be moved to the `Backup` folder on the D: drive.

6.2 Move System Files

- **Command:** `move /s [SourceFile] [Destination]`
- **Description:** Moves system files along with regular files.

Example:
bash
Copy code
```
move /s C:\Users\YourName\Documents\SystemFile.sys D:\Backup\
```

- **Result:** The system file `SystemFile.sys` will be moved to the `Backup` folder on the D: drive.

---

7. del - Delete files.

2.1 Delete Multiple Files with Wildcards

- **Command:** `del [FilePath]\[Wildcard]`
- **Description:** Deletes multiple files matching a wildcard pattern.

Example:
bash

Copy code
```
del C:\Users\YourName\Documents\*.txt
```

- Result: Deletes all `.txt` files from the `Documents` folder.

2.2 Delete All Files in a Folder

- Command: `del [FolderPath]\*.*`
- Description: Deletes all files in the specified folder, but does not delete subfolders.

Example:
bash
Copy code
```
del C:\Users\YourName\Documents\*.*
```

- Result: Deletes all files in the `Documents` folder.

---

3. Deleting Files with Confirmation

3.1 Prompt Before Deleting Files

- Command: `del /p [FilePath]`
- Description: Prompts for confirmation before deleting each file.

Example:
bash
Copy code
```
del /p C:\Users\YourName\Documents\File1.txt
```

- Result: Asks for confirmation before deleting `File1.txt`.

---

4. Deleting Read-Only Files

4.1 Force Deletion of Read-Only Files

- Command: `del /f [FilePath]`
- Description: Forces deletion of read-only files, if they exist.

Example:
bash

Copy code
```
del /f C:\Users\YourName\Documents\ReadOnlyFile.txt
```

- Result: Deletes the read-only file `ReadOnlyFile.txt` from the `Documents` folder.

---

5. Deleting Files Without Confirmation

5.1 Delete Files Without Asking for Confirmation

- Command: `del /q [FilePath]`
- Description: Deletes files quietly, without asking for confirmation.

Example:
bash
Copy code
```
del /q C:\Users\YourName\Documents\File1.txt
```

- Result: Deletes `File1.txt` without any confirmation prompt.

5.2 Delete All Files in a Folder Without Confirmation

- Command: `del /q [FolderPath]\*.*`
- Description: Deletes all files in the specified folder without confirmation.

Example:
bash
Copy code
```
del /q C:\Users\YourName\Documents\*.*
```

- Result: Deletes all files in the `Documents` folder without confirmation.

---

6. Deleting Hidden or System Files

6.1 Delete Hidden Files

- Command: `del /h [FilePath]`
- Description: Deletes hidden files along with regular files.

Example:
bash

Copy code
```
del /h C:\Users\YourName\Documents\HiddenFile.txt
```

- Result: Deletes the hidden file `HiddenFile.txt` from the `Documents` folder.

6.2 Delete System Files

- Command: `del /s [FilePath]`
- Description: Deletes system files as well as regular files.

Example:
bash
Copy code
```
del /s C:\Users\YourName\Documents\SystemFile.sys
```

- Result: Deletes the system file `SystemFile.sys` from the `Documents` folder.

---

7. Deleting Files in Subdirectories

7.1 Delete Files in All Subdirectories

- Command: `del /s [FolderPath]\*.*`
- Description: Deletes all files in the specified folder and its subdirectories.

Example:
bash
Copy code
```
del /s C:\Users\YourName\Documents\*.*
```

- Result: Deletes all files in the `Documents` folder and its subdirectories.

7.2 Delete Files in Subdirectories Without Asking for Confirmation

- Command: `del /q /s [FolderPath]\*.*`
- Description: Deletes all files in the specified folder and its subdirectories without confirmation.

Example:
bash
Copy code
```
del /q /s C:\Users\YourName\Documents\*.*
```

- **Result: Deletes all files in the `Documents` folder and its subdirectories without asking for confirmation.**

---

8. Deleting Files with Specific Extensions

8.1 Delete All Files with a Specific Extension

- **Command: `del [FolderPath]\*.ext`**
- **Description: Deletes all files with the specified extension in the given folder.**

Example:
bash
Copy code
```
del C:\Users\YourName\Documents\*.log
```

- **Result: Deletes all `.log` files from the `Documents` folder.**

---

9. Using `del` with Batch Files

9.1 Delete Files Using a Batch File

- **Command: Use `del` within a batch file (.bat) to delete files.**

Example (Batch File):
batch
Copy code
```
del C:\Users\YourName\Documents\File1.txt
```

- **Result: The batch file will delete `File1.txt` from the `Documents` folder when executed.**

8. ren - Rename files or directories

1.1 Rename a Single File

- **Command:** `ren [OldFileName] [NewFileName]`
- **Description:** Renames a single file or directory from an old name to a new name.

Example:
bash
Copy code
```
ren C:\Users\YourName\Documents\oldFile.txt newFile.txt
```

- **Result:** Renames `oldFile.txt` to `newFile.txt` in the `Documents` folder.

---

2. Rename Files with Wildcards

2.1 Rename Multiple Files with a Pattern

- **Command:** `ren [FolderPath]\[OldPattern] [NewPattern]`
- **Description:** Renames multiple files in a folder using wildcards to match a pattern.

Example:
bash
Copy code
```
ren C:\Users\YourName\Documents\*.txt *.doc
```

- **Result:** Renames all `.txt` files in the `Documents` folder to `.doc`.

2.2 Rename Files with Wildcards (Specific Part of Filename)

- **Command:** `ren [FolderPath]\[OldPattern] [NewPattern]`
- **Description:** Renames files by replacing part of the filename using wildcards.

Example:
bash
Copy code
```
ren C:\Users\YourName\Documents\*2020*.txt *2021*.txt
```

- **Result:** Renames all `.txt` files containing `2020` in their filename to `2021`.

---

3. Rename Directory or Folder

3.1 Rename a Directory

- **Command:** `ren [OldFolderName] [NewFolderName]`

- **Description: Renames a directory (folder) from an old name to a new name.**

Example:
bash
Copy code
```
ren C:\Users\YourName\Documents\oldFolder newFolder
```

- **Result: Renames the folder `oldFolder` to `newFolder` in the `Documents` directory.**

---

4. Handling Spaces in Filenames

4.1 Rename a File or Folder with Spaces in Its Name

- **Command: `ren "[Old File/Folder Name]" "[New File/Folder Name]"`**
- **Description: Renames a file or folder that contains spaces in its name. Enclose the file or folder name in quotes.**

Example:
bash
Copy code
```
ren "C:\Users\YourName\Documents\Old File.txt" "New File.txt"
```

- **Result: Renames `Old File.txt` to `New File.txt` in the `Documents` folder.**

---

5. Batch Rename Files Using a Batch File

5.1 Rename Multiple Files in a Folder

- **Command: Use `ren` in a batch file (.bat) to rename multiple files in a folder.**

Example (Batch File):
batch
Copy code
```
ren C:\Users\YourName\Documents\file1.txt file1_renamed.txt

ren C:\Users\YourName\Documents\file2.txt file2_renamed.txt
```

- **Result: Renames `file1.txt` to `file1_renamed.txt` and `file2.txt` to `file2_renamed.txt` in the `Documents` folder.**

---

6. Rename Multiple Files with Different Extensions

6.1 Rename All Files with a Specific Extension

- **Command:** `ren [FolderPath]\*.ext [NewPattern].ext`
- **Description:** Renames all files with a specific extension in the folder.

Example:
bash
Copy code
```
ren C:\Users\YourName\Documents\*.jpg *.png
```

- **Result:** Renames all `.jpg` files in the `Documents` folder to `.png`.

---

7. Rename Files by Removing Specific Characters

7.1 Remove Specific Part of Filename

- **Command:** `ren [FolderPath]\[OldPattern] [NewPattern]`
- **Description:** Removes a specific part of a filename using wildcards.
- **Example:**
  bash
  Copy code
  ```
  ren C:\Users\YourName\Documents\*old*.* *new*.*
  ```

9. `type` Command: Display the Contents of a Text File

9.1 Display the Entire Contents of a Text File

Command:
cmd
Copy code
```
type [FilePath]
```

-
- **Description:**
  Displays the complete contents of a specified text file in the Command Prompt window.

Example:
cmd

Copy code

```
type C:\Users\YourName\Documents\example.txt
```

-
- Result:
  Outputs the content of `example.txt` to the console.

9.2 Combine and Display Multiple Files

Command:
cmd
Copy code

```
type [FilePath1] [FilePath2] [FilePathN]
```

-
- Description:
  Displays the combined contents of multiple files in sequence.

Example:
cmd
Copy code

```
type file1.txt file2.txt
```

-
- Result:
  Outputs the contents of `file1.txt` followed by `file2.txt`.

9.3 Redirect Output to Another File

Command:
cmd
Copy code

```
type [FilePath] > [NewFilePath]
```

-
- Description:
  Reads the contents of a file and writes it to a new file.

Example:
cmd

Copy code
```
type oldfile.txt > newfile.txt
```

-
- Result:
  Copies the contents of `oldfile.txt` into `newfile.txt`.

9.4 Append Contents to an Existing File

Command:
cmd
Copy code
```
type [FilePath] >> [ExistingFilePath]
```

-
- Description:
  Appends the contents of one file to another file.

Example:
cmd
Copy code
```
type additional.txt >> mainfile.txt
```

-
- Result:
  Adds the content of `additional.txt` to the end of `mainfile.txt`.

---

10. find - Search for a specific string in a file.

1. Search for a Specific String in a File

Command:
cmd
Copy code
```
find "[SearchString]" [FilePath]
```

-
- Description: Finds the specified string in the given file.

Example:
cmd
Copy code

```
find "error" C:\logs\logfile.txt
```

-

---

2. Search for a String in All Files of a Folder

Command:
cmd
Copy code

```
find "[SearchString]" C:\Folder\*.*
```

-
- **Description: Searches for the string in all files within a folder.**

Example:
cmd
Copy code

```
find "error" C:\Folder\*.*
```

-

---

3. Search for a String with Case Sensitivity

Command:
cmd
Copy code

```
findstr /c:"[SearchString]" [FilePath]
```

-
- **Description: Performs a case-sensitive search for the string.**

Example:
cmd
Copy code

```
findstr /c:"ERROR" C:\logs\logfile.txt
```

-

---

4. Search for a String in All Files (Recursive Search)

Command:
cmd
Copy code

```
find "[SearchString]" /s [Directory]
```

-
- **Description: Searches in the folder and all subdirectories.**

Example:
cmd
Copy code

```
find "error" /s C:\logs
```

-

---

5. Display Only Matching Filenames

Command:
cmd
Copy code

```
findstr /m "[SearchString]" C:\logs\*.*
```

-
- **Description: Lists only the filenames that contain the matching string.**

Example:
cmd
Copy code

```
findstr /m "error" C:\logs\*.*
```

-

---

6. Search for Multiple Strings

Command:
cmd

Copy code

```
findstr "[String1]" [FilePath] | findstr "[String2]"
```

-
- Description: Finds multiple strings in a file.

Example:
cmd
Copy code

```
findstr "error" C:\logs\logfile.txt | findstr "warning"
```

-

---

7. Search for Exact Matches

Command:
cmd
Copy code

```
findstr /x "[SearchString]" [FilePath]
```

-
- Description: Searches for the exact match of the string.

Example:
cmd
Copy code

```
findstr /x "error" C:\logs\logfile.txt
```

-

---

8. Search for String in Compressed Files

Command:
cmd
Copy code

```
findstr "[SearchString]" [CompressedFile.zip]
```

-
- Description: Allows searching inside compressed files (ZIP).

Example:
cmd
Copy code

```
findstr "error" C:\logs\logfile.zip
```

-

---

9. Show Only Files with Match in Their Name

Command:
cmd
Copy code

```
dir | findstr /i "[SearchString]"
```

-
- **Description: Displays files with the search string in their names.**

Example:
cmd
Copy code

```
dir | findstr /i "error"
```

-

---

10. Search in All Files Inside a Folder (Recursive)

Command:
cmd
Copy code

```
find "[SearchString]" C:\Folder\*.*
```

-
- **Description: Searches all files within a folder recursively.**

Example:
cmd
Copy code

```
find "error" C:\Folder\*.*
```

-

---

11. Find Files Based on Modified Date

Command:
cmd
Copy code

```
dir /T:W | find "[SearchString]"
```

-
- **Description: Finds files based on their modified date.**

Example:
cmd
Copy code

```
dir /T:W | find "01/12/2023"
```

-

---

12. Display Lines Matching a Specific String

Command:
cmd
Copy code

```
findstr /n "[SearchString]" [FilePath]
```

-
- **Description: Displays the line numbers of matching lines.**

Example:
cmd
Copy code

```
findstr /n "error" C:\logs\logfile.txt
```

-

---

13. Search for Specific Keywords

Command:
cmd

Copy code
```
find "[Keyword]" [FilePath]
```

-
- Description: Finds lines that contain a specific keyword.

Example:
cmd
Copy code
```
find "success" C:\logs\logfile.txt
```

-

---

14. Search for a Specific String in Specific Files

Command:
cmd
Copy code
```
findstr "[SearchString]" C:\logs\logfile1.txt
```

-
- Description: Searches specific files for the desired string.

Example:
cmd
Copy code
```
findstr "error" C:\logs\logfile1.txt
```

-

---

15. Show Lines Matching Specific Pattern

Command:
cmd
Copy code
```
findstr /r "^.{5}$" [FilePath]
```

-
- Description: Finds lines that match a pattern.

Example:
cmd
Copy code

```
findstr /r "^.{5}$" C:\logs\logfile.txt
```

-

---

16. Display Files with Larger Content

Command:
cmd
Copy code

```
dir /S | findstr /c:"[SearchString]"
```

-
- **Description: Displays files larger than a specific size.**

Example:
cmd
Copy code

```
dir /S | findstr /c:"500MB"
```

-

---

17. Search for Specific Line Length

Command:
cmd
Copy code

```
findstr "^.{40,}" [FilePath]
```

-
- **Description: Finds lines longer than 40 characters.**

Example:
cmd
Copy code

```
findstr "^.{40,}" C:\logs\logfile.txt
```

-

---

18. Find Occurrences Using Word Boundaries

Command:
cmd
Copy code

```
findstr "\b[SearchString]\b" [FilePath]
```

-
- **Description: Finds strings using word boundaries for exact matches.**

Example:
cmd
Copy code

```
findstr "\berror\b" C:\logs\logfile.txt
```

-

---

19. Limit Search Output to First Few Lines

Command:
cmd
Copy code

```
findstr "[SearchString]" [FilePath] | head -n 10
```

-
- **Description: Limits search output to the first 10 lines.**

Example:
cmd
Copy code

```
findstr "error" C:\logs\logfile.txt | head -n 10
```

-

---

20. Search for Specific File Extension

Command:
cmd

Copy code
```
find "[SearchString]" C:\*.txt
```

-
- Description: Searches specific file types (e.g., `.txt`).

Example:
cmd
Copy code
```
find "error" C:\logs\*.txt
```

-

---

21. Display Lines Containing Any of Multiple Strings

Command:
cmd
Copy code
```
findstr "[String1]\|[String2]" [FilePath]
```

-
- Description: Displays lines matching any of the specified strings.

Example:
cmd
Copy code
```
findstr "error\|warning" C:\logs\logfile.txt
```

-

---

22. Search for Files with No Matches

Command:
cmd
Copy code
```
find /v "[SearchString]" [FilePath]
```

-
- Description: Finds files that do not contain the search string.

Example:
cmd
Copy code

```
find /v "error" C:\logs\logfile.txt
```

11. attrib - Change file attributes.

1. Change File Attributes

Command:
cmd
Copy code

```
attrib [Attribute] [FilePath]
```

-
- **Description: Changes the attributes of a file or directory. Attributes control how files are handled by the operating system, such as whether they are read-only, hidden, or system files.**

---

2. Set the Read-Only Attribute

Command:
cmd
Copy code

```
attrib +r [FilePath]
```

-
- **Description: Sets the file or directory as read-only. This prevents modifications or deletions of the file.**

Example:
cmd
Copy code

```
attrib +r C:\Documents\example.txt
```

-

---

3. Remove the Read-Only Attribute

Command:
cmd
Copy code

```
attrib -r [FilePath]
```

-
- **Description: Removes the read-only attribute, allowing the file to be edited or deleted.**

Example:
cmd
Copy code

```
attrib -r C:\Documents\example.txt
```

-

---

4. Set the Hidden Attribute

Command:
cmd
Copy code

```
attrib +h [FilePath]
```

-
- **Description: Marks the file or directory as hidden, making it not visible in normal directory listings.**

Example:
cmd
Copy code

```
attrib +h C:\Documents\example.txt
```

-

---

5. Remove the Hidden Attribute

Command:
cmd

Copy code
```
attrib -h [FilePath]
```

-
- Description: Removes the hidden attribute, making the file visible again in directory listings.

Example:
cmd
Copy code
```
attrib -h C:\Documents\example.txt
```

-

---

6. Set the System Attribute

Command:
cmd
Copy code
```
attrib +s [FilePath]
```

-
- Description: Marks the file or directory as a system file, which is typically used by the operating system and hidden from normal users.

Example:
cmd
Copy code
```
attrib +s C:\Windows\System32\example.dll
```

-

---

7. Remove the System Attribute

Command:
cmd
Copy code
```
attrib -s [FilePath]
```

-
- Description: Removes the system attribute from a file or directory.

Example:
cmd
Copy code

```
attrib -s C:\Windows\System32\example.dll
```

-

---

8. Set the Archive Attribute

Command:
cmd
Copy code

```
attrib +a [FilePath]
```

-
- **Description: Marks the file for archiving, which typically means it needs to be backed up.**

Example:
cmd
Copy code

```
attrib +a C:\Documents\example.txt
```

-

---

9. Remove the Archive Attribute

Command:
cmd
Copy code

```
attrib -a [FilePath]
```

-
- **Description: Removes the archive attribute, marking the file as not needing to be backed up.**

Example:
cmd
Copy code

```
attrib -a C:\Documents\example.txt
```

-

---

10. Set All Attributes (Read-Only, Hidden, System, Archive)

Command:
cmd
Copy code

```
attrib +r +h +s +a [FilePath]
```

-
- **Description: Sets multiple attributes for a file, making it read-only, hidden, a system file, and marked for archiving.**

Example:
cmd
Copy code

```
attrib +r +h +s +a C:\Documents\example.txt
```

-

---

11. Remove All Attributes (Read-Only, Hidden, System, Archive)

Command:
cmd
Copy code

```
attrib -r -h -s -a [FilePath]
```

-
- **Description: Removes all attributes from a file or directory, making it normal (not hidden, system, or read-only).**

Example:
cmd
Copy code

```
attrib -r -h -s -a C:\Documents\example.txt
```

-

---

12. View File Attributes

Command:
cmd
Copy code
```
attrib [FilePath]
```

-
- **Description: Displays the current attributes of a file or directory.**

Example:
cmd
Copy code
```
attrib C:\Documents\example.txt
```

12. tree - Display directory structure as a tree.

1. Basic Command Usage

Command:
cmd
Copy code
```
tree [Drive:\Path]
```

-
- **Description: Displays a graphical representation of the directory structure starting from the specified directory or drive.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents
```

-
    - **This will display the directory structure of `Documents` in a tree format.**

---

2. Display the Directory Structure with File Names

Command:
cmd
Copy code
```
tree [Drive:\Path] /f
```

-

- **Description: Displays the directory structure along with the filenames in each directory.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents /f
```

-
    - **This will show the directory structure of `Documents` with the files listed inside each directory.**

---

3. Display the Directory Structure in Color

Command:
cmd
Copy code
```
tree [Drive:\Path] /f /a
```

-
- **Description: Displays the directory structure using ASCII characters and adds color to the output.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents /f /a
```

-
    - **This shows the directory structure with files, using ASCII characters for the tree structure and color-coding the output.**

---

4. Display Only the Directory Structure (No Files)

Command:
cmd
Copy code
```
tree [Drive:\Path] /a
```

-

- **Description: Displays the directory structure without showing individual files, only directories.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents /a
```

-
    - **This shows only the directory structure without listing the files.**

---

5. Display the Directory Structure in a Specific Format (ASCII vs. Extended Characters)

Command:
cmd
Copy code
```
tree [Drive:\Path] /a
```

-
- **Description: By default, the `tree` command uses extended characters for the tree structure, but using `/a` forces it to use ASCII characters instead.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents /a
```

-

---

6. Display the Directory Structure for All Subdirectories

Command:
cmd
Copy code
```
tree [Drive:\Path] /f /a /s
```

-
- **Description: Displays the directory structure, including all subdirectories, and lists the files for each directory.**

Example:
cmd
Copy code
```
tree C:\ /f /a /s
```

-
    - **This shows the entire directory structure of `C:\`, including all subdirectories and files, using ASCII characters.**

---

7. Limit the Depth of the Directory Tree

Command:
cmd
Copy code
```
tree [Drive:\Path] /f /a /l [Level]
```

-
- **Description: Limits the depth of the tree display to a specified number of levels. The `Level` parameter defines how many levels deep the tree will be shown.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents /f /a /l 2
```

-
    - **This limits the tree to display only two levels of directories and files under `Documents`.**

---

8. Redirect Output to a File

Command:
cmd
Copy code
```
tree [Drive:\Path] > [FilePath]
```

-
- **Description: Redirects the output of the `tree` command to a text file instead of displaying it on the screen.**

Example:
cmd
Copy code

```
tree C:\Users\YourName\Documents > C:\Users\YourName\Documents\directory_structure.txt
```

-
    - **This will save the directory structure of `Documents` to a text file `directory_structure.txt`.**

---

9. Display the Directory Structure for a Remote Server (using a network drive)

Command:
cmd
Copy code

```
tree \\[ServerName]\[ShareName]
```

-
- **Description: Displays the directory structure for a network share or a remote server.**

Example:
cmd
Copy code

```
tree \\Server01\SharedDocs
```

-
    - **This will display the directory structure of the shared folder `SharedDocs` on the remote server `Server01`.**

---

10. View the Directory Tree with Directories Only (No Files)

Command:
cmd
Copy code

```
tree [Drive:\Path] /a /d
```

-

- **Description: Shows only the directories in a tree format without displaying any files.**
- **Example:**
  cmd
  Copy code
  `tree C:\Users\YourName\Documents /a /d`

13. xcopy - Extended copy command with more options

1. Basic Command Usage

Command:
cmd
Copy code
`xcopy [Source] [Destination]`

-
- **Description: Copies files or directories from the source to the destination.**

Example:
cmd
Copy code
`xcopy C:\Users\YourName\Documents D:\Backup`

-
    - **This copies the `Documents` folder from `C:\Users\YourName` to `D:\Backup`.**

---

2. Copy Files and Subdirectories

Command:
cmd
Copy code
`xcopy [Source] [Destination] /s`

-
- **Description: Copies all files and subdirectories, excluding empty directories.**

Example:
cmd
Copy code

```
xcopy C:\Users\YourName\Documents D:\Backup /s
```

-
    - **This copies the `Documents` folder and all its contents, excluding empty directories.**

---

3. Copy Files, Including Empty Subdirectories

Command:
cmd
Copy code

```
xcopy [Source] [Destination] /e
```

-
- **Description: Copies all files and subdirectories, including empty directories.**

Example:
cmd
Copy code

```
xcopy C:\Users\YourName\Documents D:\Backup /e
```

-
    - **This copies the `Documents` folder and all its contents, including empty subdirectories.**

---

4. Copy Hidden and System Files

Command:
cmd
Copy code

```
xcopy [Source] [Destination] /h
```

-
- **Description: Copies hidden and system files along with regular files.**

Example:
cmd

Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /h
```

-
    - This copies all files in `Documents`, including hidden and system files.

---

5. Copy Only Files That Are Newer

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /d
```

-
- **Description: Copies only files that are newer than the existing files in the destination.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /d
```

-
    - This copies only files from `Documents` that are newer than those already in `D:\Backup`.

---

6. Prompt Before Overwriting

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /y
```

-
- **Description: Suppresses prompting to confirm you want to overwrite an existing file. By default, `xcopy` prompts before overwriting.**

Example:
cmd

Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /y
```

-
    - This copies `Documents` to `D:\Backup` and suppresses any overwrite prompts.

---

7. Copy Files and Preserve File Attributes

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /k
```

-
- Description: Copies files and preserves their file attributes (read-only, hidden, etc.).

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /k
```

-
    - This copies the files from `Documents` to `D:\Backup` while preserving the original file attributes.

---

8. Copy Files Over a Network (Using UNC Path)

Command:
cmd
Copy code
```
xcopy [Source] \\[NetworkDrive]\[Share] [Destination]
```

-
- Description: Copies files from a local source to a remote network drive or share.

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents \\Server01\SharedDocs
```

-
    - This copies the `Documents` folder to the shared network folder `SharedDocs` on the server `Server01`.

---

9. Display Progress of File Copying

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /v
```

-
- **Description: Verifies each file as it is copied and displays the progress of copying.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /v
```

-
    - This will display the progress as files are copied from `Documents` to `D:\Backup`.

---

10. Exclude Files Based on Criteria

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /exclude:[file]
```

-
- **Description: Excludes files that match a pattern defined in the `[file]` list.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /exclude:exclude.txt
```

-

- This copies all files in `Documents` to `D:\Backup`, excluding any files listed in the `exclude.txt` file.

---

11. Display a Brief Summary

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /q
```

-
- **Description: Suppresses the display of file names being copied, showing only a summary.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /q
```

-
  - **This copies the files from `Documents` to `D:\Backup` but only shows the summary of files copied, not the names of each file.**

---

12. Copy a Directory and All Files, with Detailed Output

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /f
```

-
- **Description: Displays the full source and destination path of each file being copied.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /f
```

-

- This copies `Documents` to `D:\Backup` and displays the full path of each file as it is copied.

---

13. Copy Files with File Attributes and Ignore Errors

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /c
```

-
- **Description: Ignores errors during the copy process and continues with the next files.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /c
```

-
  - **This copies `Documents` to `D:\Backup`, and if any errors occur (such as file permission issues), it ignores them and proceeds with the rest of the files.**

---

14. Copy Files with Date/Time Criteria

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /t
```

-
- **Description: Copies the directory structure (including subdirectories) but does not copy the files.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /t
```

-

- This will copy the directory structure of `Documents` to `D:\Backup`, but no files will be copied.

---

15. Copy Files Based on File Size

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /size:[size]
```

-
- **Description: Copies files that meet the size criteria.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /size:1024
```

-
  - **This copies only files in `Documents` that are 1024 bytes (1KB) or larger to `D:\Backup`.**

---

16. Copy Files Overwriting All Files in the Destination

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /o
```

-
- **Description: Copies files and retains the original owner and permissions.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /o
```

-
  - **This copies files and ensures the original owner and permissions are kept intact during the copy process.**

14. chkdsk - Check and repair disk errors.

1. Basic Command Usage

Command:
cmd
Copy code
`chkdsk [Drive:]`

-
- **Description: Checks the specified drive for errors and reports if any are found.**

Example:
cmd
Copy code
`chkdsk C:`

-
    - **This checks the `C:` drive for disk errors and displays the results.**

---

2. Check and Repair Disk Errors

Command:
cmd
Copy code
`chkdsk [Drive:] /f`

-
- **Description: Scans for disk errors and automatically fixes any issues it detects. You may need to schedule a restart if the drive is in use.**

Example:
cmd
Copy code
`chkdsk C: /f`

-
    - **This checks the `C:` drive and fixes any errors that are found.**

---

3. Check for Bad Sectors and Repair Them

Command:
cmd
Copy code
```
chkdsk [Drive:] /r
```

-
- **Description: Scans for bad sectors on the disk and attempts to recover readable information. This option also includes the `/f` option for fixing errors.**

Example:
cmd
Copy code
```
chkdsk C: /r
```

-
    - **This checks the `C:` drive, fixes errors, and attempts to recover any readable data from bad sectors.**

---

4. Display Detailed Information About the Disk Check

Command:
cmd
Copy code
```
chkdsk [Drive:] /v
```

-
- **Description: Provides a detailed list of all files that are checked during the disk scan, showing more verbose output.**

Example:
cmd
Copy code
```
chkdsk C: /v
```

-
    - **This checks the `C:` drive and displays detailed information about the check process.**

---

5. Check for Disk Errors Without Making Changes

Command:
cmd
Copy code
```
chkdsk [Drive:] /x
```

-
- **Description: Forces the disk to dismount before checking for errors, which may make the check faster, but it could result in data loss if applications are using the drive.**

Example:
cmd
Copy code
```
chkdsk C: /x
```

-
    - **This checks the `C:` drive by first dismounting it before performing the scan.**

---

6. Check for Errors and Log Results to a File

Command:
cmd
Copy code
```
chkdsk [Drive:] /l:[Size]
```

-
- **Description: Logs the results of the `chkdsk` scan to a file, allowing you to specify the size of the log.**

Example:
cmd
Copy code
```
chkdsk C: /l:50
```

-
    - **This logs the output to a file and limits the size of the log file to 50 KB.**

---

7. Schedule a Disk Check on Next System Restart

Command:
cmd
Copy code
```
chkdsk [Drive:] /f /r /x
```

-
- **Description: Schedules a disk check for the next time the system is restarted, with automatic fixes for errors and bad sector recovery.**

Example:
cmd
Copy code
```
chkdsk C: /f /r /x
```

-
    - **This schedules a check for the `C:` drive on the next restart, fixing errors, recovering bad sectors, and dismounting the drive if needed.**

---

8. Check for File System Integrity

Command:
cmd
Copy code
```
chkdsk [Drive:] /scan
```

-
- **Description: Scans the file system for errors without requiring the system to be rebooted.**

Example:
cmd
Copy code
```
chkdsk C: /scan
```

-
    - **This scans the `C:` drive for file system errors without requiring a restart.**

---

9. Check for Errors on External or Unmounted Drives

Command:
cmd

Copy code
```
chkdsk [Drive:] /f /r
```

-
- Description: Checks and repairs errors, and checks for bad sectors on an external drive or unmounted drive.

Example:
cmd
Copy code
```
chkdsk E: /f /r
```

-
    - This checks and repairs errors and attempts to recover data from bad sectors on the `E:` drive.

---

10. Display the Status of the Last Disk Check

Command:
cmd
Copy code
```
chkdsk [Drive:] /i
```

-
- Description: Displays the status of the last check disk operation.

Example:
cmd
Copy code
```
chkdsk C: /i
```

-
    - This shows the results of the last check disk run on the `C:` drive.

---

11. Check for Errors on Remote Disk (Network Drive)

Command:
cmd
Copy code
```
chkdsk \\[RemoteDrive]\[Path] /f
```

-
- **Description: Checks for disk errors on a remote drive over the network.**

Example:
cmd
Copy code
```
chkdsk \\Server01\SharedDrive /f
```

-
    - **This checks for errors on the remote network drive `SharedDrive` on `Server01` and fixes any detected errors.**

---

12. Check for Specific File System Integrity Issues

Command:
cmd
Copy code
```
chkdsk [Drive:] /c
```

-
- **Description: Disables the checking of cycles within the folder structure. Use this when you know the file system is intact and you want to speed up the check.**

Example:
cmd
Copy code
```
chkdsk C: /c
```

-
    - **This checks the `C:` drive without checking for cycles in the folder structure.**

15. fc - Compare two files or sets of files.

1. Basic Command Usage

Command:
cmd

Copy code
```
fc [File1] [File2]
```

-
- Description: Compares two files line by line and displays any differences between them.

Example:
cmd
Copy code
```
fc file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt` and shows the lines that differ between them.

---

2. Compare Files in ASCII Mode

Command:
cmd
Copy code
```
fc /a [File1] [File2]
```

-
- Description: Compares files in ASCII mode (text files). It displays line-by-line differences between two text files.

Example:
cmd
Copy code
```
fc /a file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt` in ASCII mode and displays the differences.

---

3. Compare Files in Binary Mode

Command:
cmd

Copy code
```
fc /b [File1] [File2]
```

-
- Description: Compares files in binary mode, which is used for non-text (binary) files. It compares files byte by byte.

Example:
cmd
Copy code
```
fc /b file1.exe file2.exe
```

-
    - This compares `file1.exe` and `file2.exe` in binary mode, showing byte-by-byte differences.

---

4. Display Differences with Line Numbers

Command:
cmd
Copy code
```
fc /l [File1] [File2]
```

-
- Description: Displays the differences between files in ASCII mode, with the line numbers of each difference.

Example:
cmd
Copy code
```
fc /l file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt` in ASCII mode and shows the line numbers where differences occur.

---

5. Ignore Whitespace Differences

Command:
cmd

Copy code
```
fc /w [File1] [File2]
```

-
- Description: Ignores differences in whitespace (spaces or tabs) while comparing files.

Example:
cmd
Copy code
```
fc /w file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt`, ignoring any differences in whitespace.

---

6. Compare Multiple Files in a Directory

Command:
cmd
Copy code
```
fc [Directory1]\* [Directory2]\*
```

-
- Description: Compares multiple files in two directories.

Example:
cmd
Copy code
```
fc C:\Documents\* D:\Backup\*
```

-
    - This compares all files in `C:\Documents\` with all files in `D:\Backup\`.

---

7. Ignore Case Differences

Command:
cmd
Copy code
```
fc /i [File1] [File2]
```

-
- **Description: Compares files while ignoring case differences in letters.**

Example:
cmd
Copy code
```
fc /i file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt`, ignoring any case differences (e.g., "Hello" vs. "hello").**

---

8. Display Only the Differences (No Summary)

Command:
cmd
Copy code
```
fc /n [File1] [File2]
```

-
- **Description: Displays only the differences between files, without a summary of the comparison.**

Example:
cmd
Copy code
```
fc /n file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt`, showing only the lines that are different.**

---

9. Compare Files and Display a Summary of Differences

Command:
cmd
Copy code
```
fc /s [File1] [File2]
```

-

- **Description: Compares files and displays a summary of differences, instead of showing the entire content.**

Example:
cmd
Copy code
```
fc /s file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt` and provides a summary of the differences without showing every single change.**

---

10. Display Differences Using a Different Character for Line Separation

Command:
cmd
Copy code
```
fc /c [File1] [File2]
```

-
- **Description: Displays differences using a specified character for line separation.**

Example:
cmd
Copy code
```
fc /c file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt` and uses a custom character to separate the lines with differences.**

16. comp - Compare the contents of two files.

---

1. Basic Command Usage

Command:
cmd
Copy code
```
comp [File1] [File2]
```

-
- **Description: Compares two files byte by byte and displays the results, showing any differences.**

Example:
cmd
Copy code
```
comp file1.txt file2.txt
```

-
    - **This compares `file1.txt` with `file2.txt` and reports any byte-level differences.**

---

2. Compare Files and Display Only Differences

Command:
cmd
Copy code
```
comp /d [File1] [File2]
```

-
- **Description: Compares two files and only displays the differences, ignoring other information.**

Example:
cmd
Copy code
```
comp /d file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt`, showing only the differences between the two files.**

---

3. Display Detailed Information About Differences

Command:
cmd
Copy code
```
comp /a [File1] [File2]
```

-
- **Description: Compares the files and displays a detailed list of differences between them, including the offset and byte-by-byte differences.**

Example:
cmd
Copy code
```
comp /a file1.bin file2.bin
```

-
    - **This compares `file1.bin` and `file2.bin`, showing detailed information about their differences at the byte level.**

---

4. Compare Files and Report the First Difference Only

Command:
cmd
Copy code
```
comp /c [File1] [File2]
```

-
- **Description: Compares files and stops after displaying the first difference found.**

Example:
cmd
Copy code
```
comp /c file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt` and stops after finding and displaying the first difference.**

---

5. Compare Files and Display Only the Number of Differences

Command:
cmd

Copy code
```
comp /n [File1] [File2]
```

-
- Description: Compares two files and displays only the total number of differences.

Example:
cmd
Copy code
```
comp /n file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt` and shows only the number of byte-level differences between them.

---

6. Compare Files and Ignore Case Differences

Command:
cmd
Copy code
```
comp /i [File1] [File2]
```

-
- Description: Compares two text files while ignoring case differences (upper vs. lower case).

Example:
cmd
Copy code
```
comp /i file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt` while ignoring any case differences (e.g., "HELLO" vs. "hello").

---

7. Compare Files and Display Hexadecimal Values

Command:
cmd
Copy code
```
comp /f [File1] [File2]
```

-
- **Description: Compares the files and displays the differences using hexadecimal values for the byte comparison.**

Example:
cmd
Copy code
```
comp /f file1.bin file2.bin
```

-
    - **This compares `file1.bin` and `file2.bin`, displaying the differences in hexadecimal format.**

---

8. Compare Multiple Files in a Directory

Command:
cmd
Copy code
```
comp [Directory1]\* [Directory2]\*
```

-
- **Description: Compares multiple files in two directories and reports the differences.**

Example:
cmd
Copy code
```
comp C:\Documents\* D:\Backup\*
```

-
    - **This compares all files in `C:\Documents` with all files in `D:\Backup`.**

---

9. Display No Output If Files Are Identical

Command:
cmd
Copy code
```
comp /q [File1] [File2]
```

-

- **Description: Suppresses the output if the files being compared are identical, providing a quieter comparison.**

Example:
cmd
Copy code

```
comp /q file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt`, and if they are identical, no output is shown.**

17. robocopy - Robust file and directory copying tool.

2. Rename Files with Wildcards

2.1 Rename Multiple Files with a Pattern

- **Command: `ren [FolderPath]\[OldPattern] [NewPattern]`**
- **Description: Renames multiple files in a folder using wildcards to match a pattern.**

Example:
bash
Copy code

```
ren C:\Users\YourName\Documents\*.txt *.doc
```

-
- **Result: Renames all `.txt` files in the `Documents` folder to `.doc`.**

2.2 Rename Files with Wildcards (Specific Part of Filename)

- **Command: `ren [FolderPath]\[OldPattern] [NewPattern]`**
- **Description: Renames files by replacing part of the filename using wildcards.**

Example:
bash
Copy code

```
ren C:\Users\YourName\Documents\*2020*.txt *2021*.txt
```

-
- Result: Renames all `.txt` files containing `2020` in their filename to `2021`.

---

3. Rename Directory or Folder

3.1 Rename a Directory

- Command: `ren [OldFolderName] [NewFolderName]`
- Description: Renames a directory (folder) from an old name to a new name.

Example:
bash
Copy code
```
ren C:\Users\YourName\Documents\oldFolder newFolder
```

-
- Result: Renames the folder `oldFolder` to `newFolder` in the `Documents` directory.

---

4. Handling Spaces in Filenames

4.1 Rename a File or Folder with Spaces in Its Name

- Command: `ren "[Old File/Folder Name]" "[New File/Folder Name]"`
- Description: Renames a file or folder that contains spaces in its name. Enclose the file or folder name in quotes.

Example:
bash
Copy code
```
ren "C:\Users\YourName\Documents\Old File.txt" "New File.txt"
```

-
- Result: Renames `Old File.txt` to `New File.txt` in the `Documents` folder.

---

5. Batch Rename Files Using a Batch File

5.1 Rename Multiple Files in a Folder

- Command: Use `ren` in a batch file (.bat) to rename multiple files in a folder.

Example (Batch File):
batch
Copy code

```
ren C:\Users\YourName\Documents\file1.txt file1_renamed.txt

ren C:\Users\YourName\Documents\file2.txt file2_renamed.txt
```

-
- Result: Renames `file1.txt` to `file1_renamed.txt` and `file2.txt` to `file2_renamed.txt` in the `Documents` folder.

---

6. Rename Multiple Files with Different Extensions

6.1 Rename All Files with a Specific Extension

- Command: `ren [FolderPath]\*.ext [NewPattern].ext`
- Description: Renames all files with a specific extension in the folder.

Example:
bash
Copy code

```
ren C:\Users\YourName\Documents\*.jpg *.png
```

-
- Result: Renames all `.jpg` files in the `Documents` folder to `.png`.

---

7. Rename Files by Removing Specific Characters

7.1 Remove Specific Part of Filename

- Command: `ren [FolderPath]\[OldPattern] [NewPattern]`
- Description: Removes a specific part of a filename using wildcards.

Example:
bash
Copy code

```
ren C:\Users\YourName\Documents\*old*.* *new*.*
```

-
- Result: Renames files by replacing `old` in the filename with `new`.

---

10. find - Search for a specific string in a file.

1. Search for a Specific String in a File

Command:
cmd
Copy code

```
find "[SearchString]" [FilePath]
```

-
- **Description: Finds the specified string in the given file.**

Example:
cmd
Copy code

```
find "error" C:\logs\logfile.txt
```

-

---

2. Search for a String in All Files of a Folder

Command:
cmd
Copy code

```
find "[SearchString]" C:\Folder\*.*
```

-
- **Description: Searches for the string in all files within a folder.**

Example:
cmd
Copy code

```
find "error" C:\Folder\*.*
```

-

---

3. Search for a String with Case Sensitivity

Command:
cmd
Copy code

```
findstr /c:"[SearchString]" [FilePath]
```

-
- **Description: Performs a case-sensitive search for the string.**

Example:
cmd
Copy code

```
findstr /c:"ERROR" C:\logs\logfile.txt
```

-

---

4. Search for a String in All Files (Recursive Search)

Command:
cmd
Copy code

```
find "[SearchString]" /s [Directory]
```

-
- **Description: Searches in the folder and all subdirectories.**

Example:
cmd
Copy code

```
find "error" /s C:\logs
```

-

---

5. Display Only Matching Filenames

Command:
cmd
Copy code

```
findstr /m "[SearchString]" C:\logs\*.*
```

-
- **Description: Lists only the filenames that contain the matching string.**

Example:
cmd
Copy code

```
findstr /m "error" C:\logs\*.*
```

-

---

6. Search for Multiple Strings

Command:
cmd
Copy code

```
findstr "[String1]" [FilePath] | findstr "[String2]"
```

-
- **Description: Finds multiple strings in a file.**

Example:
cmd
Copy code

```
findstr "error" C:\logs\logfile.txt | findstr "warning"
```

-

---

7. Search for Exact Matches

Command:
cmd
Copy code

```
findstr /x "[SearchString]" [FilePath]
```

-
- **Description: Searches for the exact match of the string.**

Example:
cmd
Copy code

```
findstr /x "error" C:\logs\logfile.txt
```

-

---

8. Search for String in Compressed Files

Command:
cmd
Copy code

```
findstr "[SearchString]" [CompressedFile.zip]
```

-
- **Description: Allows searching inside compressed files (ZIP).**

Example:
cmd
Copy code

```
findstr "error" C:\logs\logfile.zip
```

-

---

9. Show Only Files with Match in Their Name

Command:
cmd
Copy code

```
dir | findstr /i "[SearchString]"
```

-
- **Description: Displays files with the search string in their names.**

Example:
cmd
Copy code

```
dir | findstr /i "error"
```

-

---

10. Search in All Files Inside a Folder (Recursive)

Command:
cmd

Copy code

```
find "[SearchString]" C:\Folder\*.*
```

-
- Description: Searches all files within a folder recursively.

Example:
cmd
Copy code

```
find "error" C:\Folder\*.*
```

-

---

11. Find Files Based on Modified Date

Command:
cmd
Copy code

```
dir /T:W | find "[SearchString]"
```

-
- Description: Finds files based on their modified date.

Example:
cmd
Copy code

```
dir /T:W | find "01/12/2023"
```

-

---

12. Display Lines Matching a Specific String

Command:
cmd
Copy code

```
findstr /n "[SearchString]" [FilePath]
```

-
- Description: Displays the line numbers of matching lines.

Example:
cmd
Copy code
```
findstr /n "error" C:\logs\logfile.txt
```

-

---

13. Search for Specific Keywords

Command:
cmd
Copy code
```
find "[Keyword]" [FilePath]
```

-
- **Description: Finds lines that contain a specific keyword.**

Example:
cmd
Copy code
```
find "success" C:\logs\logfile.txt
```

-

---

14. Search for a Specific String in Specific Files

Command:
cmd
Copy code
```
findstr "[SearchString]" C:\logs\logfile1.txt
```

-
- **Description: Searches specific files for the desired string.**

Example:
cmd
Copy code
```
findstr "error" C:\logs\logfile1.txt
```

-

---

15. Show Lines Matching Specific Pattern

Command:
cmd
Copy code

```
findstr /r "^.{5}$" [FilePath]
```

-
- **Description: Finds lines that match a pattern.**

Example:
cmd
Copy code

```
findstr /r "^.{5}$" C:\logs\logfile.txt
```

-

---

16. Display Files with Larger Content

Command:
cmd
Copy code

```
dir /S | findstr /c:"[SearchString]"
```

-
- **Description: Displays files larger than a specific size.**

Example:
cmd
Copy code

```
dir /S | findstr /c:"500MB"
```

-

---

17. Search for Specific Line Length

Command:
cmd

Copy code
```
findstr "^.{40,}" [FilePath]
```

-
- Description: Finds lines longer than 40 characters.

Example:
cmd
Copy code
```
findstr "^.{40,}" C:\logs\logfile.txt
```

-

---

18. Find Occurrences Using Word Boundaries

Command:
cmd
Copy code
```
findstr "\b[SearchString]\b" [FilePath]
```

-
- Description: Finds strings using word boundaries for exact matches.

Example:
cmd
Copy code
```
findstr "\berror\b" C:\logs\logfile.txt
```

-

---

19. Limit Search Output to First Few Lines

Command:
cmd
Copy code
```
findstr "[SearchString]" [FilePath] | head -n 10
```

-
- Description: Limits search output to the first 10 lines.

Example:
cmd
Copy code
```
findstr "error" C:\logs\logfile.txt | head -n 10
```

-

---

20. Search for Specific File Extension

Command:
cmd
Copy code
```
find "[SearchString]" C:\*.txt
```

-
- **Description: Searches specific file types (e.g., `.txt`).**

Example:
cmd
Copy code
```
find "error" C:\logs\*.txt
```

-

---

21. Display Lines Containing Any of Multiple Strings

Command:
cmd
Copy code
```
findstr "[String1]\|[String2]" [FilePath]
```

-
- **Description: Displays lines matching any of the specified strings.**

Example:
cmd
Copy code
```
findstr "error\|warning" C:\logs\logfile.txt
```

-

---

22. Search for Files with No Matches

Command:
cmd
Copy code

```
find /v "[SearchString]" [FilePath]
```

-
- **Description: Finds files that do not contain the search string.**

Example:
cmd
Copy code

```
find /v "error" C:\logs\logfile.txt
```

-

---

11. attrib - Change file attributes.

1. Change File Attributes

Command:
cmd
Copy code

```
attrib [Attribute] [FilePath]
```

-
- **Description: Changes the attributes of a file or directory. Attributes control how files are handled by the operating system, such as whether they are read-only, hidden, or system files.**

---

2. Set the Read-Only Attribute

Command:
cmd

Copy code
```
attrib +r [FilePath]
```

-
- Description: Sets the file or directory as read-only. This prevents modifications or deletions of the file.

Example:
cmd
Copy code
```
attrib +r C:\Documents\example.txt
```

-

---

3. Remove the Read-Only Attribute

Command:
cmd
Copy code
```
attrib -r [FilePath]
```

-
- Description: Removes the read-only attribute, allowing the file to be edited or deleted.

Example:
cmd
Copy code
```
attrib -r C:\Documents\example.txt
```

-

---

4. Set the Hidden Attribute

Command:
cmd
Copy code
```
attrib +h [FilePath]
```

-

- **Description: Marks the file or directory as hidden, making it not visible in normal directory listings.**

Example:
cmd
Copy code
```
attrib +h C:\Documents\example.txt
```

-

---

5. Remove the Hidden Attribute

Command:
cmd
Copy code
```
attrib -h [FilePath]
```

-
- **Description: Removes the hidden attribute, making the file visible again in directory listings.**

Example:
cmd
Copy code
```
attrib -h C:\Documents\example.txt
```

-

---

6. Set the System Attribute

Command:
cmd
Copy code
```
attrib +s [FilePath]
```

-
- **Description: Marks the file or directory as a system file, which is typically used by the operating system and hidden from normal users.**

Example:
cmd

Copy code

```
attrib +s C:\Windows\System32\example.dll
```

-

---

7. Remove the System Attribute

Command:
cmd
Copy code

```
attrib -s [FilePath]
```

-
- **Description: Removes the system attribute from a file or directory.**

Example:
cmd
Copy code

```
attrib -s C:\Windows\System32\example.dll
```

-

---

8. Set the Archive Attribute

Command:
cmd
Copy code

```
attrib +a [FilePath]
```

-
- **Description: Marks the file for archiving, which typically means it needs to be backed up.**

Example:
cmd
Copy code

```
attrib +a C:\Documents\example.txt
```

-

---

9. Remove the Archive Attribute

Command:
cmd
Copy code

```
attrib -a [FilePath]
```

-
- **Description: Removes the archive attribute, marking the file as not needing to be backed up.**

Example:
cmd
Copy code

```
attrib -a C:\Documents\example.txt
```

-

---

10. Set All Attributes (Read-Only, Hidden, System, Archive)

Command:
cmd
Copy code

```
attrib +r +h +s +a [FilePath]
```

-
- **Description: Sets multiple attributes for a file, making it read-only, hidden, a system file, and marked for archiving.**

Example:
cmd
Copy code

```
attrib +r +h +s +a C:\Documents\example.txt
```

-

---

11. Remove All Attributes (Read-Only, Hidden, System, Archive)

Command:
cmd

Copy code
```
attrib -r -h -s -a [FilePath]
```

-
- Description: Removes all attributes from a file or directory, making it normal (not hidden, system, or read-only).

Example:
cmd
Copy code
```
attrib -r -h -s -a C:\Documents\example.txt
```

-

---

12. View File Attributes

Command:
cmd
Copy code
```
attrib [FilePath]
```

-
- Description: Displays the current attributes of a file or directory.

Example:
cmd
Copy code
```
attrib C:\Documents\example.txt
```

-

---

12. tree - Display directory structure as a tree.

1. Basic Command Usage

Command:
cmd
Copy code
```
tree [Drive:\Path]
```

-
- **Description: Displays a graphical representation of the directory structure starting from the specified directory or drive.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents
```

-
    - **This will display the directory structure of `Documents` in a tree format.**

---

2. Display the Directory Structure with File Names

Command:
cmd
Copy code
```
tree [Drive:\Path] /f
```

-
- **Description: Displays the directory structure along with the filenames in each directory.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents /f
```

-
    - **This will show the directory structure of `Documents` with the files listed inside each directory.**

---

3. Display the Directory Structure in Color

Command:
cmd
Copy code
```
tree [Drive:\Path] /f /a
```

-

- **Description: Displays the directory structure using ASCII characters and adds color to the output.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents /f /a
```

-
    - **This shows the directory structure with files, using ASCII characters for the tree structure and color-coding the output.**

---

4. Display Only the Directory Structure (No Files)

Command:
cmd
Copy code
```
tree [Drive:\Path] /a
```

-
- **Description: Displays the directory structure without showing individual files, only directories.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents /a
```

-
    - **This shows only the directory structure without listing the files.**

---

5. Display the Directory Structure in a Specific Format (ASCII vs. Extended Characters)

Command:
cmd
Copy code
```
tree [Drive:\Path] /a
```

-

- **Description: By default, the `tree` command uses extended characters for the tree structure, but using `/a` forces it to use ASCII characters instead.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents /a
```

-

---

6. Display the Directory Structure for All Subdirectories

Command:
cmd
Copy code
```
tree [Drive:\Path] /f /a /s
```

-
- **Description: Displays the directory structure, including all subdirectories, and lists the files for each directory.**

Example:
cmd
Copy code
```
tree C:\ /f /a /s
```

-
    - **This shows the entire directory structure of `C:\`, including all subdirectories and files, using ASCII characters.**

---

7. Limit the Depth of the Directory Tree

Command:
cmd
Copy code
```
tree [Drive:\Path] /f /a /l [Level]
```

-
- **Description: Limits the depth of the tree display to a specified number of levels. The `Level` parameter defines how many levels deep the tree will be shown.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents /f /a /l 2
```

-
    - **This limits the tree to display only two levels of directories and files under `Documents`.**

---

8. Redirect Output to a File

Command:
cmd
Copy code
```
tree [Drive:\Path] > [FilePath]
```

-
- **Description: Redirects the output of the `tree` command to a text file instead of displaying it on the screen.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents > C:\Users\YourName\Documents\directory_structure.txt
```

-
    - **This will save the directory structure of `Documents` to a text file `directory_structure.txt`.**

---

9. Display the Directory Structure for a Remote Server (using a network drive)

Command:
cmd
Copy code
```
tree \\[ServerName]\[ShareName]
```

-

- **Description: Displays the directory structure for a network share or a remote server.**

Example:
cmd
Copy code
```
tree \\Server01\SharedDocs
```

-
    - **This will display the directory structure of the shared folder `SharedDocs` on the remote server `Server01`.**

---

10. View the Directory Tree with Directories Only (No Files)

Command:
cmd
Copy code
```
tree [Drive:\Path] /a /d
```

-
- **Description: Shows only the directories in a tree format without displaying any files.**

Example:
cmd
Copy code
```
tree C:\Users\YourName\Documents /a /d
```

-

---

13. xcopy - Extended copy command with more options.

1. Basic Command Usage

Command:
cmd
Copy code
```
xcopy [Source] [Destination]
```

-
- **Description: Copies files or directories from the source to the destination.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup
```

-
    - **This copies the `Documents` folder from `C:\Users\YourName` to `D:\Backup`.**

---

2. Copy Files and Subdirectories

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /s
```

-
- **Description: Copies all files and subdirectories, excluding empty directories.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /s
```

-
    - **This copies the `Documents` folder and all its contents, excluding empty directories.**

---

3. Copy Files, Including Empty Subdirectories

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /e
```

-
- **Description: Copies all files and subdirectories, including empty directories.**

Example:
cmd
Copy code

```
xcopy C:\Users\YourName\Documents D:\Backup /e
```

-
    - **This copies the `Documents` folder and all its contents, including empty subdirectories.**

---

4. Copy Hidden and System Files

Command:
cmd
Copy code

```
xcopy [Source] [Destination] /h
```

-
- **Description: Copies hidden and system files along with regular files.**

Example:
cmd
Copy code

```
xcopy C:\Users\YourName\Documents D:\Backup /h
```

-
    - **This copies all files in `Documents`, including hidden and system files.**

---

5. Copy Only Files That Are Newer

Command:
cmd
Copy code

```
xcopy [Source] [Destination] /d
```

-
- **Description: Copies only files that are newer than the existing files in the destination.**

Example:
cmd

Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /d
```

-
    - This copies only files from `Documents` that are newer than those already in `D:\Backup`.

---

6. Prompt Before Overwriting

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /y
```

-
- Description: Suppresses prompting to confirm you want to overwrite an existing file. By default, `xcopy` prompts before overwriting.

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /y
```

-
    - This copies `Documents` to `D:\Backup` and suppresses any overwrite prompts.

---

7. Copy Files and Preserve File Attributes

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /k
```

-
- Description: Copies files and preserves their file attributes (read-only, hidden, etc.).

Example:
cmd

Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /k
```

-
    - This copies the files from `Documents` to `D:\Backup` while preserving the original file attributes.

---

8. Copy Files Over a Network (Using UNC Path)

Command:
cmd
Copy code
```
xcopy [Source] \\[NetworkDrive]\[Share] [Destination]
```

-
- **Description: Copies files from a local source to a remote network drive or share.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents \\Server01\SharedDocs
```

-
    - This copies the `Documents` folder to the shared network folder `SharedDocs` on the server `Server01`.

---

9. Display Progress of File Copying

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /v
```

-
- **Description: Verifies each file as it is copied and displays the progress of copying.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /v
```

-
    - This will display the progress as files are copied from `Documents` to `D:\Backup`.

---

10. Exclude Files Based on Criteria

Command:
cmd
Copy code
`xcopy [Source] [Destination] /exclude:[file]`

-
- **Description: Excludes files that match a pattern defined in the `[file]` list.**

Example:
cmd
Copy code
`xcopy C:\Users\YourName\Documents D:\Backup /exclude:exclude.txt`

-
    - **This copies all files in `Documents` to `D:\Backup`, excluding any files listed in the `exclude.txt` file.**

---

11. Display a Brief Summary

Command:
cmd
Copy code
`xcopy [Source] [Destination] /q`

-
- **Description: Suppresses the display of file names being copied, showing only a summary.**

Example:
cmd
Copy code
`xcopy C:\Users\YourName\Documents D:\Backup /q`

-

- This copies the files from `Documents` to `D:\Backup` but only shows the summary of files copied, not the names of each file.

---

12. Copy a Directory and All Files, with Detailed Output

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /f
```

-
- **Description: Displays the full source and destination path of each file being copied.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /f
```

-
  - This copies `Documents` to `D:\Backup` and displays the full path of each file as it is copied.

---

13. Copy Files with File Attributes and Ignore Errors

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /c
```

-
- **Description: Ignores errors during the copy process and continues with the next files.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /c
```

-

- This copies `Documents` to `D:\Backup`, and if any errors occur (such as file permission issues), it ignores them and proceeds with the rest of the files.

---

14. Copy Files with Date/Time Criteria

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /t
```

-
- **Description: Copies the directory structure (including subdirectories) but does not copy the files.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /t
```

-
    - **This will copy the directory structure of `Documents` to `D:\Backup`, but no files will be copied.**

---

15. Copy Files Based on File Size

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /size:[size]
```

-
- **Description: Copies files that meet the size criteria.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /size:1024
```

-

- This copies only files in `Documents` that are 1024 bytes (1KB) or larger to `D:\Backup`.

---

16. Copy Files Overwriting All Files in the Destination

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /o
```

-
- **Description: Copies files and retains the original owner and permissions.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents D:\Backup /o
```

-
  - **This copies files and ensures the original owner and permissions are kept intact during the copy process.**

---

14. chkdsk - Check and repair disk errors.

1. Basic Command Usage

Command:
cmd
Copy code
```
chkdsk [Drive:]
```

-
- **Description: Checks the specified drive for errors and reports if any are found.**

Example:
cmd
Copy code
```
chkdsk C:
```

-

- This checks the `C:` drive for disk errors and displays the results.

---

2. Check and Repair Disk Errors

Command:
cmd
Copy code
```
chkdsk [Drive:] /f
```

-
- **Description: Scans for disk errors and automatically fixes any issues it detects. You may need to schedule a restart if the drive is in use.**

Example:
cmd
Copy code
```
chkdsk C: /f
```

-
  - **This checks the `C:` drive and fixes any errors that are found.**

---

3. Check for Bad Sectors and Repair Them

Command:
cmd
Copy code
```
chkdsk [Drive:] /r
```

-
- **Description: Scans for bad sectors on the disk and attempts to recover readable information. This option also includes the `/f` option for fixing errors.**

Example:
cmd
Copy code
```
chkdsk C: /r
```

-
  - **This checks the `C:` drive, fixes errors, and attempts to recover any readable data from bad sectors.**

---

4. Display Detailed Information About the Disk Check

Command:
cmd
Copy code
```
chkdsk [Drive:] /v
```

-
- **Description: Provides a detailed list of all files that are checked during the disk scan, showing more verbose output.**

Example:
cmd
Copy code
```
chkdsk C: /v
```

-
    - **This checks the `C:` drive and displays detailed information about the check process.**

---

5. Check for Disk Errors Without Making Changes

Command:
cmd
Copy code
```
chkdsk [Drive:] /x
```

-
- **Description: Forces the disk to dismount before checking for errors, which may make the check faster, but it could result in data loss if applications are using the drive.**

Example:
cmd
Copy code
```
chkdsk C: /x
```

-
    - **This checks the `C:` drive by first dismounting it before performing the scan.**

---

6. Check for Errors and Log Results to a File

Command:
cmd
Copy code
```
chkdsk [Drive:] /l:[Size]
```

-
- **Description: Logs the results of the `chkdsk` scan to a file, allowing you to specify the size of the log.**

Example:
cmd
Copy code
```
chkdsk C: /l:50
```

-
    - **This logs the output to a file and limits the size of the log file to 50 KB.**

---

7. Schedule a Disk Check on Next System Restart

Command:
cmd
Copy code
```
chkdsk [Drive:] /f /r /x
```

-
- **Description: Schedules a disk check for the next time the system is restarted, with automatic fixes for errors and bad sector recovery.**

Example:
cmd
Copy code
```
chkdsk C: /f /r /x
```

-
    - **This schedules a check for the `C:` drive on the next restart, fixing errors, recovering bad sectors, and dismounting the drive if needed.**

---

8. Check for File System Integrity

Command:
cmd
Copy code
```
chkdsk [Drive:] /scan
```

-
- **Description: Scans the file system for errors without requiring the system to be rebooted.**

Example:
cmd
Copy code
```
chkdsk C: /scan
```

-
    - **This scans the `C:` drive for file system errors without requiring a restart.**

---

9. Check for Errors on External or Unmounted Drives

Command:
cmd
Copy code
```
chkdsk [Drive:] /f /r
```

-
- **Description: Checks and repairs errors, and checks for bad sectors on an external drive or unmounted drive.**

Example:
cmd
Copy code
```
chkdsk E: /f /r
```

-
    - **This checks and repairs errors and attempts to recover data from bad sectors on the `E:` drive.**

---

10. Display the Status of the Last Disk Check

Command:
cmd

Copy code
```
chkdsk [Drive:] /i
```

-
- Description: Displays the status of the last check disk operation.

Example:
cmd
Copy code
```
chkdsk C: /i
```

-
    - This shows the results of the last check disk run on the `C:` drive.

---

11. Check for Errors on Remote Disk (Network Drive)

Command:
cmd
Copy code
```
chkdsk \\[RemoteDrive]\[Path] /f
```

-
- Description: Checks for disk errors on a remote drive over the network.

Example:
cmd
Copy code
```
chkdsk \\Server01\SharedDrive /f
```

-
    - This checks for errors on the remote network drive `SharedDrive` on `Server01` and fixes any detected errors.

---

12. Check for Specific File System Integrity Issues

Command:
cmd
Copy code
```
chkdsk [Drive:] /c
```

-

- **Description: Disables the checking of cycles within the folder structure. Use this when you know the file system is intact and you want to speed up the check.**

Example:
cmd
Copy code
```
chkdsk C: /c
```

-
    - **This checks the `C:` drive without checking for cycles in the folder structure.**

---

15. fc - Compare two files or sets of files.

1. Basic Command Usage

Command:
cmd
Copy code
```
fc [File1] [File2]
```

-
- **Description: Compares two files line by line and displays any differences between them.**

Example:
cmd
Copy code
```
fc file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt` and shows the lines that differ between them.**

---

2. Compare Files in ASCII Mode

Command:
cmd
Copy code
```
fc /a [File1] [File2]
```

-
- **Description: Compares files in ASCII mode (text files). It displays line-by-line differences between two text files.**

Example:
cmd
Copy code
```
fc /a file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt` in ASCII mode and displays the differences.**

---

3. Compare Files in Binary Mode

Command:
cmd
Copy code
```
fc /b [File1] [File2]
```

-
- **Description: Compares files in binary mode, which is used for non-text (binary) files. It compares files byte by byte.**

Example:
cmd
Copy code
```
fc /b file1.exe file2.exe
```

-
    - **This compares `file1.exe` and `file2.exe` in binary mode, showing byte-by-byte differences.**

---

4. Display Differences with Line Numbers

Command:
cmd
Copy code
```
fc /l [File1] [File2]
```

-

- **Description: Displays the differences between files in ASCII mode, with the line numbers of each difference.**

Example:
cmd
Copy code
```
fc /l file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt` in ASCII mode and shows the line numbers where differences occur.**

---

5. Ignore Whitespace Differences

Command:
cmd
Copy code
```
fc /w [File1] [File2]
```

-
- **Description: Ignores differences in whitespace (spaces or tabs) while comparing files.**

Example:
cmd
Copy code
```
fc /w file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt`, ignoring any differences in whitespace.**

---

6. Compare Multiple Files in a Directory

Command:
cmd
Copy code
```
fc [Directory1]\* [Directory2]\*
```

-
- **Description: Compares multiple files in two directories.**

Example:
cmd
Copy code
```
fc C:\Documents\* D:\Backup\*
```

-
    - **This compares all files in `C:\Documents\` with all files in `D:\Backup\`.**

---

7. Ignore Case Differences

Command:
cmd
Copy code
```
fc /i [File1] [File2]
```

-
- **Description: Compares files while ignoring case differences in letters.**

Example:
cmd
Copy code
```
fc /i file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt`, ignoring any case differences (e.g., "Hello" vs. "hello").**

---

8. Display Only the Differences (No Summary)

Command:
cmd
Copy code
```
fc /n [File1] [File2]
```

-
- **Description: Displays only the differences between files, without a summary of the comparison.**

Example:
cmd

Copy code
```
fc /n file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt`, showing only the lines that are different.

---

9. Compare Files and Display a Summary of Differences

Command:
cmd
Copy code
```
fc /s [File1] [File2]
```

-
- Description: Compares files and displays a summary of differences, instead of showing the entire content.

Example:
cmd
Copy code
```
fc /s file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt` and provides a summary of the differences without showing every single change.

---

10. Display Differences Using a Different Character for Line Separation

Command:
cmd
Copy code
```
fc /c [File1] [File2]
```

-
- Description: Displays differences using a specified character for line separation.

Example:
cmd
Copy code
```
fc /c file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt` and uses a custom character to separate the lines with differences.

---

16. comp - Compare the contents of two files.

1. Basic Command Usage

Command:
cmd
Copy code
```
comp [File1] [File2]
```

-
- **Description: Compares two files byte by byte and displays the results, showing any differences.**

Example:
cmd
Copy code
```
comp file1.txt file2.txt
```

-
    - This compares `file1.txt` with `file2.txt` and reports any byte-level differences.

---

2. Compare Files and Display Only Differences

Command:
cmd
Copy code
```
comp /d [File1] [File2]
```

-
- **Description: Compares two files and only displays the differences, ignoring other information.**

Example:
cmd

Copy code
```
comp /d file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt`, showing only the differences between the two files.

---

3. Display Detailed Information About Differences

Command:
cmd
Copy code
```
comp /a [File1] [File2]
```

-
- Description: Compares the files and displays a detailed list of differences between them, including the offset and byte-by-byte differences.

Example:
cmd
Copy code
```
comp /a file1.bin file2.bin
```

-
    - This compares `file1.bin` and `file2.bin`, showing detailed information about their differences at the byte level.

---

4. Compare Files and Report the First Difference Only

Command:
cmd
Copy code
```
comp /c [File1] [File2]
```

-
- Description: Compares files and stops after displaying the first difference found.

Example:
cmd
Copy code
```
comp /c file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt` and stops after finding and displaying the first difference.

---

5. Compare Files and Display Only the Number of Differences

Command:
cmd
Copy code
```
comp /n [File1] [File2]
```

-
- **Description: Compares two files and displays only the total number of differences.**

Example:
cmd
Copy code
```
comp /n file1.txt file2.txt
```

-
    - This compares `file1.txt` and `file2.txt` and shows only the number of byte-level differences between them.

---

6. Compare Files and Ignore Case Differences

Command:
cmd
Copy code
```
comp /i [File1] [File2]
```

-
- **Description: Compares two text files while ignoring case differences (upper vs. lower case).**

Example:
cmd
Copy code
```
comp /i file1.txt file2.txt
```

-

- This compares `file1.txt` and `file2.txt` while ignoring any case differences (e.g., "HELLO" vs. "hello").

---

7. Compare Files and Display Hexadecimal Values

Command:
cmd
Copy code
```
comp /f [File1] [File2]
```

-
- **Description: Compares the files and displays the differences using hexadecimal values for the byte comparison.**

Example:
cmd
Copy code
```
comp /f file1.bin file2.bin
```

-
  - This compares `file1.bin` and `file2.bin`, displaying the differences in hexadecimal format.

---

8. Compare Multiple Files in a Directory

Command:
cmd
Copy code
```
comp [Directory1]\* [Directory2]\*
```

-
- **Description: Compares multiple files in two directories and reports the differences.**

Example:
cmd
Copy code
```
comp C:\Documents\* D:\Backup\*
```

-
  - This compares all files in `C:\Documents` with all files in `D:\Backup`.

---

9. Display No Output If Files Are Identical

Command:
cmd
Copy code
```
comp /q [File1] [File2]
```

-
- **Description: Suppresses the output if the files being compared are identical, providing a quieter comparison.**

Example:
cmd
Copy code
```
comp /q file1.txt file2.txt
```

-
    - **This compares `file1.txt` and `file2.txt`, and if they are identical, no output is shown.**

---

17. robocopy - Robust file and directory copying tool.

1. Basic Command Usage

Command:
cmd
Copy code
```
robocopy [Source] [Destination]
```

-
- **Description: Copies all files from the source directory to the destination directory.**

Example:
cmd
Copy code
```
robocopy C:\Source D:\Backup
```

-

- This command copies all files from `C:\Source` to `D:\Backup`.

---

2. Copy Files and Subdirectories

Command:
cmd
Copy code

```
robocopy [Source] [Destination] /E
```

-
- **Description: Copies all files and subdirectories, including empty directories.**

Example:
cmd
Copy code

```
robocopy C:\Source D:\Backup /E
```

-
  - This copies all files and subdirectories from `C:\Source` to `D:\Backup`, including any empty directories.

---

3. Copy Only Newer Files

Command:
cmd
Copy code

```
robocopy [Source] [Destination] /XO
```

-
- **Description: Copies only the files that are newer in the source directory than in the destination directory.**

Example:
cmd
Copy code

```
robocopy C:\Source D:\Backup /XO
```

-
  - This copies only the newer files from `C:\Source` to `D:\Backup`, skipping older files.

---

4. Retry on Failure

Command:
cmd
Copy code
`robocopy [Source] [Destination] /R:[NumberOfRetries]`

-
- **Description: Sets the number of retries on failed copies. By default, `robocopy` retries 1 million times, but you can specify a custom number.**

Example:
cmd
Copy code
`robocopy C:\Source D:\Backup /R:5`

-
    - **This will retry copying up to 5 times if a file cannot be copied.**

---

5. Skip Existing Files

Command:
cmd
Copy code
`robocopy [Source] [Destination] /XC /XN /XO`

-
- **Description: Skips files that already exist at the destination. It checks for files that are the same in both locations and skips copying them.**

Example:
cmd
Copy code
`robocopy C:\Source D:\Backup /XC /XN /XO`

-
    - **This command will skip existing, newer, or unchanged files when copying from `C:\Source` to `D:\Backup`.**

---

6. Copy Files with Permissions

Command:
cmd
Copy code
```
robocopy [Source] [Destination] /COPYALL
```

-
- **Description: Copies all file attributes and permissions (including timestamps and ACLs).**

Example:
cmd
Copy code
```
robocopy C:\Source D:\Backup /COPYALL
```

-
    - **This copies all files from `C:\Source` to `D:\Backup`, including all permissions, attributes, and timestamps.**

---

7. Mirror Directories

Command:
cmd
Copy code
```
robocopy [Source] [Destination] /MIR
```

-
- **Description: Mirrors a directory, copying all files and subdirectories from the source to the destination, and deleting files at the destination that are no longer in the source.**

Example:
cmd
Copy code
```
robocopy C:\Source D:\Backup /MIR
```

-
    - **This creates an exact mirror of `C:\Source` in `D:\Backup`, deleting any files in the destination that do not exist in the source.**

---

8. Limit the Bandwidth Used During Copying

Command:
cmd
Copy code

```
robocopy [Source] [Destination] /IPG:[Delay]
```

-
- **Description: Sets the inter-packet gap (IPG) in milliseconds to limit the bandwidth used during file copying.**

Example:
cmd
Copy code

```
robocopy C:\Source D:\Backup /IPG:50
```

-
    - **This limits the copying speed by introducing a 50 millisecond delay between packets.**

---

9. Exclude Specific Files

Command:
cmd
Copy code

```
robocopy [Source] [Destination] /XF [FileName]
```

-
- **Description: Excludes specific files from being copied.**

Example:
cmd
Copy code

```
robocopy C:\Source D:\Backup /XF *.log
```

-
    - **This copies everything from `C:\Source` to `D:\Backup`, but skips any `.log` files.**

---

10. Exclude Specific Directories

Command:
cmd
Copy code
`robocopy [Source] [Destination] /XD [DirectoryName]`

-
- **Description: Excludes specific directories from being copied.**

Example:
cmd
Copy code
`robocopy C:\Source D:\Backup /XD C:\Source\Temp`

-
    - **This copies everything from `C:\Source` to `D:\Backup`, excluding the `Temp` directory.**

---

11. Perform a Dry Run

Command:
cmd
Copy code
`robocopy [Source] [Destination] /L`

-
- **Description: Simulates the copy operation without actually making any changes (dry run).**

Example:
cmd
Copy code
`robocopy C:\Source D:\Backup /L`

-
    - **This simulates the copying process from `C:\Source` to `D:\Backup` but does not actually copy any files.**

---

12. Copy Files with Logging

Command:
cmd

Copy code

```
robocopy [Source] [Destination] /LOG:[LogFile]
```

-
- Description: Logs the output of the copying process to a specified log file.

Example:
cmd
Copy code

```
robocopy C:\Source D:\Backup /LOG:C:\BackupLog.txt
```

-
    - This copies from `C:\Source` to `D:\Backup` and logs the details to `C:\BackupLog.txt`.

---

13. Use Robocopy for Network Copying

Command:
cmd
Copy code

```
robocopy \\[NetworkSource] \\[NetworkDestination]
```

-
- Description: Uses `robocopy` to copy files over a network to a shared folder.

Example:
cmd
Copy code

```
robocopy \\Server\Share D:\Backup
```

-
    - This copies files from the network share `\\Server\Share` to the `D:\Backup` directory.

---

14. Copy Empty Directories Only

Command:
cmd
Copy code

```
robocopy [Source] [Destination] /E /LEV:1
```

-
- Description: Copies only empty directories from the source to the destination.

Example:
cmd
Copy code
```
robocopy C:\Source D:\Backup /E /LEV:1
```

-
    - This copies only empty directories from `C:\Source` to `D:\Backup`.

18. sfc - System File Checker to repair corrupted system files.

---

1. Basic Command Usage

Command:
cmd
Copy code
```
sfc /scannow
```

-
- Description: Scans all protected system files and repairs any corrupted or missing files.

Example:
cmd
Copy code
```
sfc /scannow
```

-
    - This command scans all system files for integrity issues and attempts to fix any detected corruption.

---

2. Check System Files Without Repairing

Command:
cmd
Copy code

```
sfc /verifyonly
```

-
- **Description: Scans protected system files but does not perform any repair actions, useful for checking the integrity of files without making changes.**

Example:
cmd
Copy code

```
sfc /verifyonly
```

-
    - **This command checks the system files for integrity but does not attempt any repairs.**

---

3. Scan a Specific File

Command:
cmd
Copy code

```
sfc /scanfile=[Path to File]
```

-
- **Description: Scans a specific file for integrity issues. This command allows you to check the status of a specific system file rather than scanning the entire system.**

Example:
cmd
Copy code

```
sfc /scanfile=C:\Windows\System32\kernel32.dll
```

-
    - **This scans the file `kernel32.dll` in the `System32` folder for corruption.**

---

4. Set the Log File Location

Command:
cmd

Copy code
```
sfc /offbootdir=[Boot Drive] /offwindir=[Windows Directory]
```

-
- Description: Specifies the boot directory and Windows directory when running `sfc` on an offline system (such as when using Windows recovery environment).

Example:
cmd
Copy code
```
sfc /offbootdir=D:\ /offwindir=D:\Windows
```

-
    - This command runs `sfc` on an offline Windows installation located on drive `D:`.

---

5. Check and Repair System Files Without a Restart

Command:
cmd
Copy code
```
sfc /scanfile=[FilePath] /offbootdir=[BootDrive]
/offwindir=[WindowsDir]
```

-
- Description: This command is used to check and repair system files on an offline system without the need for a restart.

Example:
cmd
Copy code
```
sfc /scanfile=C:\Windows\System32\user32.dll /offbootdir=C:\
/offwindir=C:\Windows
```

-
    - This checks and repairs the `user32.dll` file located in `C:\Windows\System32`.

---

6. View Detailed Logs of the Scan

Command:
cmd
Copy code
`sfc /log`

-
- **Description: Displays a detailed log of the results from the `sfc` scan, which can help identify issues that the tool has found and fixed.**

Example:
cmd
Copy code
`sfc /log`

-
    - **This command displays a detailed log of the `sfc` scan results, useful for advanced troubleshooting.**

---

7. Troubleshoot Problems Using the Windows Recovery Environment

Command:
cmd
Copy code
`sfc /scannow /offbootdir=[Boot Drive] /offwindir=[Windows Directory]`

-
- **Description: Runs `sfc` in the Windows Recovery Environment to scan and repair system files without starting the Windows operating system.**

Example:
cmd
Copy code
`sfc /scannow /offbootdir=C:\ /offwindir=C:\Windows`

-
    - **This command scans and repairs system files from the Windows Recovery Environment.**

---

8. Repair System Files Using DISM

Sometimes `sfc` might not be able to repair the system files if the source files are also corrupted. In such cases, the `DISM` (Deployment Imaging Service and Management Tool) can be used to restore the files.

Command:
cmd
Copy code
```
dism /online /cleanup-image /restorehealth
```

-
- Description: This command uses DISM to repair the Windows image and restore missing or corrupted system files before running `sfc`.

Example:
cmd
Copy code
```
dism /online /cleanup-image /restorehealth
```

-
    - This command repairs the system image and prepares it for `sfc` to complete its repair process.

---

9. Repair Missing or Corrupt Files Using `sfc` After DISM

After using `DISM`, you can run `sfc` to complete the file repair process:

Command:
cmd
Copy code
```
sfc /scannow
```

-
- Description: Scans and repairs system files after running `DISM` to restore the Windows image.

Example:
cmd
Copy code
```
sfc /scannow
```

-

- This command runs the System File Checker to fix any remaining file integrity issues after `DISM` has restored the system image.

---

10. Repair Windows Modules with SFC

Command:
cmd
Copy code

```
sfc /scannow /offbootdir=[Boot Drive] /offwindir=[Windows Directory] /modules
```

-
- **Description: Scans and repairs specific Windows modules, ensuring that all components are working correctly.**

Example:
cmd
Copy code

```
sfc /scannow /offbootdir=C:\ /offwindir=C:\Windows /modules
```

-
    - **This command repairs specific modules of Windows during the `sfc` scan.**

19. findstr - Search for specific strings in files.

1. Basic Command Usage

Command:
cmd
Copy code

```
findstr [String] [FilePath]
```

-
- **Description: Searches for the specified string in a file and displays the matching lines.**

Example:
cmd
Copy code

```
findstr "error" C:\Logs\logfile.txt
```

-
    - This command searches for the string "error" in `logfile.txt` located in the `C:\Logs\` directory and displays all lines containing the word "error."

---

2. Search Multiple Strings

Command:
cmd
Copy code
```
findstr [String1] [String2] [FilePath]
```

-
- **Description: Searches for multiple strings in a file. Any line containing at least one of the strings will be returned.**

Example:
cmd
Copy code
```
findstr "error" "warning" C:\Logs\logfile.txt
```

-
    - This searches for lines containing either "error" or "warning" in `logfile.txt`.

---

3. Search Using Regular Expressions

Command:
cmd
Copy code
```
findstr /r [RegularExpression] [FilePath]
```

-
- **Description: Uses regular expressions to search for patterns in a file.**

Example:
cmd
Copy code
```
findstr /r "^[A-Za-z]*" C:\Logs\logfile.txt
```

-

- This searches for lines starting with any alphabetic characters in `logfile.txt`.

---

4. Display Line Numbers of Matches

Command:
cmd
Copy code
```
findstr /n [String] [FilePath]
```

-
- **Description: Displays the line numbers along with the lines that match the search string.**

Example:
cmd
Copy code
```
findstr /n "error" C:\Logs\logfile.txt
```

-
  - **This command shows all lines containing "error" along with their line numbers in `logfile.txt`.**

---

5. Search for Whole Words Only

Command:
cmd
Copy code
```
findstr /w [String] [FilePath]
```

-
- **Description: Searches for whole words only, ignoring partial matches.**

Example:
cmd
Copy code
```
findstr /w "error" C:\Logs\logfile.txt
```

-
  - **This command searches for the exact word "error" in `logfile.txt` and does not match words like "errors" or "erroneous."**

---

6. Case-Insensitive Search

Command:
cmd
Copy code
```
findstr /i [String] [FilePath]
```

-
- **Description: Searches for the string in a case-insensitive manner.**

Example:
cmd
Copy code
```
findstr /i "ERROR" C:\Logs\logfile.txt
```

-
    - **This searches for "ERROR", "error", or any case variation of "error" in `logfile.txt`.**

---

7. Exclude Certain Strings (Invert Match)

Command:
cmd
Copy code
```
findstr /v [String] [FilePath]
```

-
- **Description: Excludes lines that contain the specified string.**

Example:
cmd
Copy code
```
findstr /v "error" C:\Logs\logfile.txt
```

-
    - **This command displays all lines in `logfile.txt` that do not contain the word "error."**

---

8. Search for Strings in Multiple Files

Command:
cmd
Copy code

```
findstr [String] [FolderPath]\*.txt
```

-
- **Description: Searches for a string in all text files in a specified folder.**

Example:
cmd
Copy code

```
findstr "critical" C:\Logs\*.txt
```

-
    - **This searches for the string "critical" in all `.txt` files within the `C:\Logs\` directory.**

---

9. Search for Strings Across All Files in a Directory

Command:
cmd
Copy code

```
findstr /s [String] [FolderPath]\*
```

-
- **Description: Searches for a string in all files within a directory and its subdirectories.**

Example:
cmd
Copy code

```
findstr /s "error" C:\Logs\*
```

-
    - **This command searches for "error" in all files under the `C:\Logs\` directory, including subdirectories.**

---

10. Match Full Lines Only

Command:
cmd

Copy code
```
findstr /x [String] [FilePath]
```

-
- Description: Searches for the exact match of the entire line.

Example:
cmd
Copy code
```
findstr /x "This is an error" C:\Logs\logfile.txt
```

-
    - This searches for lines that are exactly "This is an error" in `logfile.txt`.

---

11. Search for Multiple Strings in Multiple Files

Command:
cmd
Copy code
```
findstr /c:"[String1]" /c:"[String2]" [FolderPath]\*.log
```

-
- Description: Searches for multiple specific strings in multiple files using the `/c:` option to treat the strings as literal patterns.

Example:
cmd
Copy code
```
findstr /c:"timeout" /c:"failure" C:\Logs\*.log
```

-
    - This command searches for both "timeout" and "failure" in all `.log` files in the `C:\Logs\` directory.

---

12. Display Only Matching Strings

Command:
cmd
Copy code
```
findstr /o [String] [FilePath]
```

-
- **Description: Displays only the matching string in the file, along with its offset (position) in the line.**

Example:
cmd
Copy code
```
findstr /o "error" C:\Logs\logfile.txt
```

-
    - **This command displays the position of the word "error" in each matching line from `logfile.txt`.**

20. more - Display the contents of a text file one page at a time.

1. Basic Command Usage

Command:
cmd
Copy code
```
more [FilePath]
```

-
- **Description: Displays the content of a file, one page at a time. Press `Enter` to advance one line at a time or `Space` to move to the next page.**

Example:
cmd
Copy code
```
more C:\Logs\logfile.txt
```

-
    - **This command shows the contents of `logfile.txt` in the `C:\Logs\` directory one page at a time.**

---

2. Display the Content of Multiple Files

Command:
cmd

Copy code
```
more [File1] [File2] ...
```

-
- Description: Displays the contents of multiple files, one page at a time.

Example:
cmd
Copy code
```
more C:\Logs\logfile1.txt C:\Logs\logfile2.txt
```

-
    - This command shows the contents of both `logfile1.txt` and `logfile2.txt` one page at a time.

---

3. Navigate Through the File

While using `more`, you can use the following keys to navigate:

- `Enter`: Display one line at a time.
- `Space`: Display one page at a time.
- `Q`: Quit and exit the `more` command.
- `/`: Search for a string in the file (e.g., `/error` to find "error").
- `B`: Go back one page.

---

4. View Long Files Line-by-Line

Command:
cmd
Copy code
```
more /e [FilePath]
```

-
- Description: Displays the contents of the file one line at a time, waiting for user input before showing each line.

Example:
cmd
Copy code
```
more /e C:\Logs\logfile.txt
```

-
    - This command will display the file content one line at a time. You will need to press `Enter` to display the next line.

---

5. Display File Content with Line Numbers

Command:
cmd
Copy code
```
more /n [FilePath]
```

-
- **Description: Displays the contents of the file with line numbers at the beginning of each line.**

Example:
cmd
Copy code
```
more /n C:\Logs\logfile.txt
```

-
    - This command will display `logfile.txt` with line numbers, making it easier to reference specific lines.

---

6. View the Contents of a File Without Pausing

Command:
cmd
Copy code
```
type [FilePath] | more
```

-
- **Description: Uses the `type` command to display the contents of the file and pipes it to `more` for paging, especially useful when the file is long.**

Example:
cmd
Copy code
```
type C:\Logs\logfile.txt | more
```

-
    - This command displays the file content page by page, even if the `type` command itself would display the entire file at once.

---

7. Skip to the End of the File

Command:
cmd
Copy code
```
more +[Number] [FilePath]
```

-
- **Description: Starts displaying the file from the specified line number. Use this option to skip the first few lines and begin from a specific point in the file.**

Example:
cmd
Copy code
```
more +10 C:\Logs\logfile.txt
```

-
    - This command starts showing the file content from the 10th line onwards.

---

8. Pause After Each Screenful

Command:
cmd
Copy code
```
more /p [FilePath]
```

-
- **Description: Pauses after each screenful of text, allowing the user to press `Enter` or `Space` to continue.**

Example:
cmd
Copy code
```
more /p C:\Logs\logfile.txt
```

-

- This command pauses after each screenful, waiting for the user to continue to the next page.

---

9. Search for Text While Viewing

Command:
cmd
Copy code

```
more /s [FilePath]
```

-
- **Description: Allows you to search for a specified string while viewing the file.**

Example:
cmd
Copy code

```
more /s "error" C:\Logs\logfile.txt
```

-
  - **This command allows searching for the word "error" while viewing the file content.**

---

10. Customize the Paging Behavior

Command:
cmd
Copy code

```
more /c [FilePath]
```

-
- **Description: Clears the screen before displaying each page of the file content.**

Example:
cmd
Copy code

```
more /c C:\Logs\logfile.txt
```

-
  - **This command clears the screen each time it displays a new page of content.**

21. sort - Sort the contents of a text file.

1. Basic Command Usage

Command:
cmd
Copy code
`sort [FilePath]`

-
- **Description: Sorts the lines in the specified text file in ascending order (alphabetically or numerically based on the content).**

Example:
cmd
Copy code
`sort C:\Documents\data.txt`

-
    - **This command sorts the contents of `data.txt` in ascending order and displays the output.**

---

2. Sort in Reverse Order (Descending)

Command:
cmd
Copy code
`sort /r [FilePath]`

-
- **Description: Sorts the lines of the text file in reverse (descending) order.**

Example:
cmd
Copy code
`sort /r C:\Documents\data.txt`

-
    - **This command sorts the contents of `data.txt` in descending order.**

---

3. Sort Without Removing Duplicate Lines

Command:
cmd
Copy code
```
sort /u [FilePath]
```

-
- **Description: Removes duplicate lines while sorting the file contents.**

Example:
cmd
Copy code
```
sort /u C:\Documents\data.txt
```

-
    - **This command sorts the file contents and removes any duplicate lines, showing only unique entries.**

---

4. Sort Output by Column

Command:
cmd
Copy code
```
sort /+n [FilePath]
```

-
- **Description: Sorts the file content starting from the nth character of each line. Useful for sorting based on columns.**

Example:
cmd
Copy code
```
sort /+5 C:\Documents\data.txt
```

-
    - **This command sorts the contents of `data.txt`, starting from the 5th character of each line.**

---

5. Sort the Contents of Standard Input (Piped Input)

Command:
cmd
Copy code
```
echo [Text] | sort
```

-
- **Description: Sorts the input provided via the command line or piped input.**

Example:
cmd
Copy code
```
echo Apple Banana Orange | sort
```

-
    - **This command sorts the words "Apple", "Banana", and "Orange" in ascending order and displays the result: "Apple Banana Orange".**

---

6. Redirect Sorted Output to a New File

Command:
cmd
Copy code
```
sort [FilePath] > [NewFilePath]
```

-
- **Description: Redirects the sorted output to a new file, rather than displaying it on the screen.**

Example:
cmd
Copy code
```
sort C:\Documents\data.txt > C:\Documents\sorted_data.txt
```

-
    - **This command sorts the contents of `data.txt` and saves the sorted output to `sorted_data.txt`.**

---

7. Sort by Specific Fields in the File (Delimited Data)

Command:
cmd
Copy code
```
sort /t[Delimiter] /+n [FilePath]
```

-
- **Description: Sorts a file with a specified delimiter (e.g., comma or tab) starting from the nth column.**

Example:
cmd
Copy code
```
sort /t, /+2 C:\Documents\data.csv
```

-
    - **This command sorts the `data.csv` file based on the second column, assuming a comma delimiter.**

---

8. Sort with Case Insensitivity

Command:
cmd
Copy code
```
sort /f [FilePath]
```

-
- **Description: Sorts the file contents in a case-insensitive manner, so "apple" and "Apple" will be treated as equivalent.**

Example:
cmd
Copy code
```
sort /f C:\Documents\data.txt
```

-
    - **This command sorts the contents of `data.txt` without considering case sensitivity.**

---

9. View Sorted Output in Real-Time Using Piping

Command:
cmd
Copy code
```
[Command] | sort
```

-
- **Description: Sorts the output of any command in real-time. You can pipe any command's output into `sort` to sort it.**

Example:
cmd
Copy code
```
dir | sort
```

-
    - **This command lists the directory contents and sorts them alphabetically.**

---

10. View Sorted Content with Line Numbers

Command:
cmd
Copy code
```
sort [FilePath] | nl
```

-
- **Description: Sorts the file content and displays line numbers along with the sorted output.**

Example:
cmd
Copy code
```
sort C:\Documents\data.txt | nl
```

-
    - **This command sorts the file `data.txt` and displays each line with a line number.**

22. xcopy /e - Copy directories and subdirectories, including empty ones.

1. Basic Command Usage

Command:
cmd
Copy code
`xcopy [Source] [Destination] /e`

-
- **Description: Copies all directories and subdirectories, including empty directories, from the source to the destination path.**

Example:
cmd
Copy code
`xcopy C:\Users\YourName\Documents\* D:\Backup\ /e`

-
    - **This command copies all files and subdirectories from `Documents` to `D:\Backup`, including any empty subdirectories.**

---

2. Copy Directories and Subdirectories, Suppressing Confirmation

Command:
cmd
Copy code
`xcopy [Source] [Destination] /e /y`

-
- **Description: Copies directories and subdirectories (including empty ones) and automatically confirms overwriting of files without prompting.**

Example:
cmd
Copy code
`xcopy C:\Users\YourName\Documents\* D:\Backup\ /e /y`

-

- This command copies all files and directories, including empty directories, and automatically overwrites any existing files in the destination without confirmation.

---

3. Copy Directories and Subdirectories with File Attributes

Command:
cmd
Copy code

```
xcopy [Source] [Destination] /e /a
```

-
- **Description: Copies directories and subdirectories, including empty ones, and preserves the file attributes (such as read-only, hidden, system).**

Example:
cmd
Copy code

```
xcopy C:\Users\YourName\Documents\* D:\Backup\ /e /a
```

-
    - **This command copies all files, subdirectories, and empty directories from `Documents` to `D:\Backup`, preserving the file attributes.**

---

4. Copy Specific File Types with Subdirectories

Command:
cmd
Copy code

```
xcopy [Source] [Destination] /e *.txt
```

-
- **Description: Copies only specific file types (in this case `.txt` files) from the source directory, including any empty subdirectories.**

Example:
cmd
Copy code

```
xcopy C:\Users\YourName\Documents\*.txt D:\Backup\ /e
```

-

- This command copies only `.txt` files from the `Documents` directory, along with any empty subdirectories, to `D:\Backup`.

---

5. Exclude Certain Files During Copying

Command:
cmd
Copy code

```
xcopy [Source] [Destination] /e /exclude:[FileWithExcludedPatterns]
```

-
- **Description: Copies directories and subdirectories, including empty ones, but excludes files that match the patterns specified in the exclude file.**

Example:
cmd
Copy code

```
xcopy C:\Users\YourName\Documents\* D:\Backup\ /e /exclude:exclude.txt
```

-
  - **This command copies all files and directories, including empty ones, except for files matching the patterns listed in `exclude.txt`.**

---

6. Copy Files and Directories with Verification

Command:
cmd
Copy code

```
xcopy [Source] [Destination] /e /v
```

-
- **Description: Copies directories and subdirectories, including empty ones, and verifies that the files are copied correctly.**

Example:
cmd
Copy code

```
xcopy C:\Users\YourName\Documents\* D:\Backup\ /e /v
```

-

- This command copies files and subdirectories, including empty directories, from `Documents` to `D:\Backup` and verifies that each file is copied correctly.

---

7. Copy Files and Directories, Including Hidden Files

Command:
cmd
Copy code
```
xcopy [Source] [Destination] /e /h
```

-
- **Description: Copies all files, directories, and subdirectories, including hidden files and folders, from the source to the destination.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents\* D:\Backup\ /e /h
```

-
  - **This command copies all files, hidden files, subdirectories, and empty directories from `Documents` to `D:\Backup`.**

---

8. Copy Multiple Source Directories to One Destination

Command:
cmd
Copy code
```
xcopy [Source1] [Source2] [Destination] /e
```

-
- **Description: Copies multiple source directories and subdirectories (including empty ones) into one destination directory.**

Example:
cmd
Copy code
```
xcopy C:\Users\YourName\Documents\* C:\Users\YourName\Pictures\*
D:\Backup\ /e
```

-
    - This command copies the contents of both `Documents` and `Pictures` directories (including empty subdirectories) to the `D:\Backup` directory.

23. compact - Compress or decompress files on an NTFS partition.

1. Compress Files

Command:
cmd
Copy code
`compact [File/Directory Path] /c`

-
- **Description: Compresses the specified file or directory on an NTFS partition.**

Example:
cmd
Copy code
`compact C:\Users\YourName\Documents\LargeFile.txt /c`

-
    - This command compresses `LargeFile.txt` in the `Documents` folder, reducing its disk usage on an NTFS partition.

---

2. Decompress Files

Command:
cmd
Copy code
`compact [File/Directory Path] /u`

-
- **Description: Decompresses the specified file or directory that was previously compressed.**

Example:
cmd
Copy code

```
compact C:\Users\YourName\Documents\CompressedFile.txt /u
```

-
    - **This command decompresses `CompressedFile.txt` in the `Documents` folder, restoring its original size.**

---

3. Display Compression Status

Command:
cmd
Copy code

```
compact [File/Directory Path]
```

-
- **Description: Displays the current compression status of a specified file or directory. It shows whether the file is compressed or uncompressed.**

Example:
cmd
Copy code

```
compact C:\Users\YourName\Documents\LargeFile.txt
```

-
    - **This command displays whether `LargeFile.txt` is currently compressed or not.**

---

4. Compress All Files in a Directory and Subdirectories

Command:
cmd
Copy code

```
compact [Directory Path] /c /s
```

-
- **Description: Compresses all files in the specified directory and all its subdirectories.**

Example:
cmd
Copy code

```
compact C:\Users\YourName\Documents\ /c /s
```

-
    - **This command compresses all files in the `Documents` folder and its subdirectories.**

---

5. Decompress All Files in a Directory and Subdirectories

Command:
cmd
Copy code

```
compact [Directory Path] /u /s
```

-
- **Description: Decompresses all files in the specified directory and all its subdirectories.**

Example:
cmd
Copy code

```
compact C:\Users\YourName\Documents\ /u /s
```

-
    - **This command decompresses all files in the `Documents` folder and its subdirectories.**

---

6. Show All Files with Compression Status in a Directory

Command:
cmd
Copy code

```
compact [Directory Path] /s /q
```

-
- **Description: Displays the compression status of all files in the specified directory and its subdirectories without making any changes.**

Example:
cmd
Copy code

```
compact C:\Users\YourName\Documents\ /s /q
```

-
    - **This command shows the compression status of all files in the `Documents` folder and its subdirectories without modifying them.**

---

7. Compress Only Files That Are Not Already Compressed

Command:
cmd
Copy code

```
compact [Directory Path] /c /a
```

-
- **Description: Compresses files in the specified directory and subdirectories, but only those that are not already compressed.**

Example:
cmd
Copy code

```
compact C:\Users\YourName\Documents\ /c /a
```

-
    - **This command compresses only the uncompressed files in the `Documents` folder and its subdirectories.**

---

8. Display Compression Information for a File

Command:
cmd
Copy code

```
compact /i [File Path]
```

-
- **Description: Displays detailed compression information for a specified file.**

Example:
cmd

Copy code
```
compact /i C:\Users\YourName\Documents\LargeFile.txt
```

-
    - This command provides detailed information about the compression status of `LargeFile.txt`.

---

9. Compress Files with Specific File Extensions

Command:
cmd
Copy code
```
compact [Directory Path] /c *.txt
```

-
- Description: Compresses only files with a specific file extension (e.g., `.txt`) in the specified directory.

Example:
cmd
Copy code
```
compact C:\Users\YourName\Documents\ /c *.txt
```

-
    - This command compresses only `.txt` files in the `Documents` folder.

---

10. Show the File Compression Ratio

Command:
cmd
Copy code
```
compact /r [File Path]
```

-
- Description: Displays the compression ratio for a compressed file, showing how much space was saved.

Example:
cmd

Copy code
```
compact /r C:\Users\YourName\Documents\LargeFile.txt
```

-
    - This command shows the compression ratio for `LargeFile.txt`, indicating the amount of space saved by compression.

24. xcaccls - Backup and restore NTFS permissions.

1. Backup NTFS Permissions

Command:
cmd
Copy code
```
xcacls [File/Folder Path] /backup:[Backup File Path]
```

-
- Description: Backs up the NTFS permissions of a specified file or folder to a backup file.

Example:
cmd
Copy code
```
xcacls C:\Users\YourName\Documents\ /backup:C:\Backup\permissions.bak
```

-
    - This command backs up the NTFS permissions of the `Documents` folder to `permissions.bak` in the `C:\Backup` directory.

---

2. Restore NTFS Permissions from a Backup

Command:
cmd
Copy code
```
xcacls [File/Folder Path] /restore:[Backup File Path]
```

-
- Description: Restores NTFS permissions to a file or folder from a previously created backup file.

Example:
cmd
Copy code
```
xcacls C:\Users\YourName\Documents\ /restore:C:\Backup\permissions.bak
```

-
    - This command restores the NTFS permissions of the `Documents` folder from the `permissions.bak` backup file.

---

3. Display Current NTFS Permissions

Command:
cmd
Copy code
```
xcacls [File/Folder Path]
```

-
- Description: Displays the current NTFS permissions of a specified file or folder.

Example:
cmd
Copy code
```
xcacls C:\Users\YourName\Documents\
```

-
    - This command displays the current NTFS permissions for the `Documents` folder.

---

4. Grant Permissions

Command:
cmd
Copy code
```
xcacls [File/Folder Path] /grant:[UserName]:[Permission]
```

-
- Description: Grants specific permissions to a user or group for a file or folder.

Example:
cmd

Copy code
```
xcacls C:\Users\YourName\Documents\ /grant:JohnDoe:F
```

-
    - This command grants full control (`F`) to the user `JohnDoe` for the `Documents` folder.

---

5. Revoke Permissions

Command:
cmd
Copy code
```
xcacls [File/Folder Path] /remove:[UserName]
```

-
- **Description: Removes the specified user's permissions from a file or folder.**

Example:
cmd
Copy code
```
xcacls C:\Users\YourName\Documents\ /remove:JohnDoe
```

-
    - This command removes all permissions for the user `JohnDoe` from the `Documents` folder.

---

6. Set Permissions Recursively

Command:
cmd
Copy code
```
xcacls [Folder Path] /t /grant:[UserName]:[Permission]
```

-
- **Description: Grants specific permissions recursively to all files and subfolders within a folder.**

Example:
cmd

Copy code
```
xcacls C:\Users\YourName\Documents\ /t /grant:JohnDoe:F
```

-
    - This command grants full control (`F`) to `JohnDoe` for the `Documents` folder and all its subdirectories and files.

---

7. Remove Inherited Permissions

Command:
cmd
Copy code
```
xcacls [File/Folder Path] /inheritance:r
```

-
- **Description: Removes inherited permissions from a file or folder. After this, only explicit permissions will apply.**

Example:
cmd
Copy code
```
xcacls C:\Users\YourName\Documents\ /inheritance:r
```

-
    - This command removes inherited permissions from the `Documents` folder.

---

8. Replace Permissions with a Backup

Command:
cmd
Copy code
```
xcacls [File/Folder Path] /replace:[Backup File Path]
```

-
- **Description: Replaces the current NTFS permissions on a file or folder with the permissions from a backup file.**

Example:
cmd
Copy code
```
xcacls C:\Users\YourName\Documents\ /replace:C:\Backup\permissions.bak
```

-
    - This command replaces the current permissions of the `Documents` folder with those from `permissions.bak`.

---

9. Grant Permissions to a Group

Command:
cmd
Copy code
```
xcacls [File/Folder Path] /grant:[GroupName]:[Permission]
```

-
- **Description: Grants specific permissions to a group for a file or folder.**

Example:
cmd
Copy code
```
xcacls C:\Users\YourName\Documents\ /grant:AdminGroup:F
```

-
    - This command grants full control (`F`) to the `AdminGroup` for the `Documents` folder.

---

10. Backup Permissions for All Files in a Directory

Command:
cmd
Copy code
```
xcacls [Directory Path] /backup:[Backup File Path] /t
```

-
- **Description: Backs up the NTFS permissions of all files and subdirectories within a directory.**

Example:
cmd
Copy code
```
xcacls C:\Users\YourName\Documents\ /backup:C:\Backup\permissions.bak /t
```

-
    - This command backs up the permissions for all files and subfolders within the `Documents` directory to `permissions.bak`.

25. subst - Associate a drive letter with a directory.

Command:

cmd

Copy code

```
subst [Drive Letter]: [Directory Path]
```

Description:
Associates a drive letter with a specified directory, making the directory appear as if it were a separate drive. This is useful for creating virtual drives for easy access to directories.

Example:

cmd

Copy code

```
subst X: C:\Users\YourName\Documents
```

This command associates the `X:` drive with the `Documents` folder located at `C:\Users\YourName\Documents`. Now, you can access that folder via the `X:` drive.

25.1. Remove a Virtual Drive

Command:

cmd

Copy code

```
subst [Drive Letter]: /d
```

Description:
Removes the virtual drive associated with the specified drive letter.

Example:

cmd

Copy code

```
subst X: /d
```

This command removes the virtual drive `X:`, disassociating it from the folder it was pointing to.

26. cipher - Display or alter file encryption on NTFS volumes.

---

1. Overview of the `cipher` Command

The `cipher` command is a tool used in Windows to manage the encryption of files and directories. It can encrypt and decrypt files, view the encryption status, securely erase data, and create backups of encryption certificates and keys.

---

2. Syntax

bash

Copy code

```
cipher [options] [drive:][path]
```

- **[options]: Various command options that specify what actions you want to perform.**
- **[drive:][path]: Path to the file, directory, or drive where encryption is applied.**

---

3. Command Options and Their Functions

a. `/E`

- **Description: Encrypts the specified file or directory.**
- **Use Case: Encrypt files and folders to protect sensitive data.**

Example:
bash
Copy code

```
cipher /E C:\Users\YourName\Documents\file.txt
```

-

b. `/D`

- **Description: Decrypts the specified file or directory.**
- **Use Case: Decrypt files or folders that were previously encrypted.**

Example:
bash
Copy code

```
cipher /D C:\Users\YourName\Documents\file.txt
```

-

c. `/S:<directory>`

- **Description: Encrypts all subdirectories within the specified directory.**
- **Use Case: Encrypt all files and subdirectories under a given directory.**

Example:
bash
Copy code

```
cipher /E /S:C:\Users\YourName\Documents
```

-

d. `/F`

- **Description: Forces the encryption or decryption of files that are already encrypted or decrypted.**
- **Use Case: Re-encrypt files that are already encrypted.**

Example:
bash
Copy code

```
cipher /E /F C:\Users\YourName\Documents
```

-

e. `/R:<filename>`

- **Description: Creates a recovery agent certificate and key.**
- **Use Case: Create a recovery certificate and key that can be used to recover encrypted files.**

Example:
bash
Copy code

```
cipher /R:RecoveryAgent
```

-

f. `/X`

- **Description: Back up the encryption certificate and key.**
- **Use Case: Save the encryption certificate and key for later use or recovery.**

Example:
bash
Copy code

```
cipher /X C:\Backup
```

-

g. `/U`

- **Description: Updates the encryption on all files and directories in the specified directory.**
- **Use Case: Ensure that all files in a directory are properly encrypted.**

Example:
bash

Copy code

```
cipher /U C:\Users\YourName\Documents
```

-

h. `/W:<folder>`

- Description: Overwrites unused disk space to securely erase deleted files.
- Use Case: Securely wipe free space to make sure deleted files cannot be recovered.

Example:
bash
Copy code

```
cipher /W:C:\Users\YourName\Documents
```

-

i. `/Q`

- Description: Suppresses the display of the status messages. Quiet mode.
- Use Case: Use this option when you want to run the command without showing the progress or results.

Example:
bash
Copy code

```
cipher /E /Q C:\Users\YourName\Documents\file.txt
```

-

j. `/A`

- Description: Displays all files in the directory, including encrypted files.
- Use Case: View both encrypted and unencrypted files in a directory.

Example:
bash
Copy code

```
cipher /A C:\Users\YourName\Documents
```

-

---

4. Detailed Use Cases and Examples

a. Encrypt a File

To encrypt a specific file, use the `/E` option followed by the file path.

Example:
bash
Copy code

```
cipher /E C:\Users\YourName\Documents\file.txt
```

-
    - **Explanation: This encrypts the file `file.txt` in the specified path.**

b. Decrypt a File

To decrypt a previously encrypted file, use the `/D` option followed by the file path.

Example:
bash
Copy code

```
cipher /D C:\Users\YourName\Documents\file.txt
```

-
    - **Explanation: This decrypts the file `file.txt`.**

c. Encrypt All Files in a Folder

To encrypt a directory and all of its subdirectories, use the `/E` option with `/S`.

Example:
bash
Copy code

```
cipher /E /S:C:\Users\YourName\Documents
```

-
    - **Explanation: Encrypts all files and subdirectories under `Documents`.**

d. Force Encryption of Already Encrypted Files

If you want to re-encrypt a file that is already encrypted, you can use the `/F` option.

Example:
bash

Copy code
```
cipher /E /F C:\Users\YourName\Documents\file.txt
```

-
    - Explanation: Forces the encryption of `file.txt`, even if it's already encrypted.

e. Create a Backup of the Encryption Certificate

To create a backup of the encryption certificate, use the `/X` option.

Example:
bash
Copy code
```
cipher /X C:\Backup
```

-
    - Explanation: Backs up the encryption certificate and key to the specified folder.

f. Securely Wipe Free Space

To securely erase deleted files and make them unrecoverable, use the `/W` option.

Example:
bash
Copy code
```
cipher /W:C:\Users\YourName\Documents
```

-
    - Explanation: Overwrites the unused disk space in `Documents` to prevent file recovery.

28. openfiles - Display or disconnect open shared files and folders.

1. Overview of the `openfiles` Command

The `openfiles` command allows users to display information about open shared files, as well as disconnect users who have files open on the system. It can be used to troubleshoot file access issues or disconnect unwanted users from shared resources.

---

2. Syntax

bash

Copy code

```
openfiles [option] [parameters]
```

- **[option]: Specifies the action you want to perform.**
- **[parameters]: Additional arguments that define the target files or user actions.**

---

3. Command Options and Their Functions

a. `/query`

- **Description: Displays a list of open shared files and folders, including the file ID, user, and access information.**
- **Use Case: View which files are currently open on the system and by which users.**

Example:
bash
Copy code

```
openfiles /query
```

-

b. `/disconnect`

- **Description: Disconnects an open file or folder that is currently being accessed by a user.**
- **Use Case: Disconnect a user from a shared file or folder to resolve issues or lockouts.**

Example:
bash

Copy code
```
openfiles /disconnect /id 100
```

-

c. `/query /fo [format]`

- Description: Displays the open files in a specific format, such as table, list, or CSV.
- Use Case: Customizes the output format for better readability or data processing.

Example:
bash
Copy code
```
openfiles /query /fo table
```

-

d. `/query /v`

- Description: Displays additional details, such as the path and type of access (read or write).
- Use Case: View detailed information about the open files.

Example:
bash
Copy code
```
openfiles /query /v
```

-

e. `/disconnect /id [fileID]`

- Description: Disconnects a file based on its ID.
- Use Case: Disconnect a specific open file based on its ID to free resources or resolve conflicts.

Example:
bash
Copy code
```
openfiles /disconnect /id 125
```

-

f. `/help`

- Description: Displays help information for the `openfiles` command.
- Use Case: Get more information about how to use the `openfiles` command.

Example:
bash
Copy code
```
openfiles /help
```

-

---

30. ipconfig - Display network configuration information.

1. Overview of the `ipconfig` Command

The `ipconfig` command is used to display the current network configuration of a computer's network interfaces. It provides valuable information for troubleshooting connectivity issues, checking IP address assignments, and managing network adapters.

---

2. Syntax

bash

Copy code

```
ipconfig [option] [parameters]
```

- [option]: Specifies the action you want to perform (e.g., display information, release/renew IP, etc.).
- [parameters]: Additional arguments that refine the command (e.g., specific network adapter details).

---

3. Command Options and Their Functions

a. `/all`

- **Description: Displays detailed information about all network interfaces on the system, including physical (MAC) addresses, DHCP status, DNS servers, and more.**
- **Use Case: Useful for troubleshooting and obtaining detailed network configuration data.**

Example:
bash
Copy code
```
ipconfig /all
```

-

b. `/release`

- **Description: Releases the current IP address configuration for the specified network adapter.**
- **Use Case: Used to disconnect from the network and drop the current IP configuration.**

Example:
bash
Copy code
```
ipconfig /release
```

-

c. `/renew`

- **Description: Renews the IP address for the specified network adapter.**
- **Use Case: Typically used after releasing the IP to obtain a new IP address from a DHCP server.**

Example:
bash
Copy code
```
ipconfig /renew
```

-

d. `/flushdns`

- **Description: Clears the DNS resolver cache. This can resolve issues where outdated or incorrect DNS entries are causing network problems.**

- **Use Case: Clears the DNS cache to ensure the system uses updated DNS information.**

Example:
bash
Copy code
```
ipconfig /flushdns
```

-

e. `/registerdns`

- **Description: Refreshes the DHCP lease and registers the computer's name with DNS.**
- **Use Case: Used when needing to update DNS records on the network.**

Example:
bash
Copy code
```
ipconfig /registerdns
```

-

f. `/showclassid`

- **Description: Displays the DHCP Class ID for a specified adapter.**
- **Use Case: Used for troubleshooting DHCP issues.**

Example:
bash
Copy code
```
ipconfig /showclassid "Ethernet"
```

-

g. `/setclassid`

- **Description: Configures a DHCP Class ID for a specified network adapter.**
- **Use Case: Used when manually setting the DHCP Class ID on a network adapter.**

Example:
bash
Copy code
```
ipconfig /setclassid "Ethernet" 1
```

-

h. `/displaydns`

- **Description: Displays the current DNS resolver cache contents.**
- **Use Case: Useful for checking the entries cached on the system by the DNS resolver.**

Example:
bash
Copy code
```
ipconfig /displaydns
```

---

31. ping - Test network connectivity.

a. Basic Network Connectivity Test

- **Description: Verifies if a system is reachable by pinging a target host.**

Example:
bash
Copy code
```
ping google.com
```

-

b. Continuous Ping

- **Description: Continuously pings a host until manually stopped (Ctrl + C).**

Example:
bash
Copy code
```
ping -t google.com
```

-

c. Specifying the Number of Echo Requests

- **Description: Sends a specific number of Echo Requests to the target host.**

Example:
bash

Copy code

```
ping -n 5 google.com
```

-

d. Customizing Packet Size

- Description: Customizes the size of the packets being sent.

Example:
bash
Copy code

```
ping -l 1500 google.com
```

-

e. Using IPv6 for Ping

- Description: Forces the ping to use IPv6 instead of IPv4.

Example:
bash
Copy code

```
ping -6 google.com
```

-

f. Resolving Hostnames from IP

- Description: Resolves and shows the hostname for the given IP address.

Example:
bash
Copy code

```
ping -a 8.8.8.8
```

-

g. Setting the "Don't Fragment" Flag

- Description: Ensures the packet is not fragmented during transmission.
- Example:
  bash
  Copy code

  ```
  ping -f google.com
  ```

32. tracert - Trace the route to a remote host.

1. Overview of the `tracert` Command

What is the `tracert` Command?

The `tracert` command traces the route that data packets follow from the source computer to a destination host, showing each router (hop) along the way and measuring the round-trip time for each hop. This information can be valuable for diagnosing network issues, such as delays or packet loss.

---

2. Syntax of `tracert` Command

General Syntax

bash

Copy code

```
tracert [hostname or IP address] [options]
```

- **[hostname or IP address]: The domain name (e.g., google.com) or IP address (e.g., 8.8.8.8) of the target host.**
- **[options]: Additional flags or parameters to modify the behavior of the command.**

---

3. Common Options for `tracert` Command

a. `-d` - Do Not Resolve Hostnames

- **Description: Prevents the command from resolving IP addresses to hostnames, which speeds up the trace.**

Example:
bash
Copy code

```
tracert -d google.com
```

-

b. `-h [max_hops]` - Set Maximum Hops

- **Description: Specifies the maximum number of hops (routers) to trace before stopping.**

Example:
bash
Copy code
```
tracert -h 15 google.com
```

-

c. `-w [timeout]` - Set Timeout for Each Reply

- **Description: Sets the timeout value (in milliseconds) for each hop before it is considered a failure.**

Example:
bash
Copy code
```
tracert -w 1000 google.com
```

-

d. `-4` - Force IPv4

- **Description: Forces the trace route to use IPv4, even if the destination host supports IPv6.**

Example:
bash
Copy code
```
tracert -4 google.com
```

-

e. `-6` - Force IPv6

- **Description: Forces the trace route to use IPv6, even if the destination host supports IPv4.**

Example:
bash

Copy code

```
tracert -6 google.com
```

-

---

33. netstat - Display network statistics.

1. Overview of the `netstat` Command

What is the `netstat` Command?

The `netstat` command provides information about network connections (both incoming and outgoing), listening ports, and other network statistics. It can be used to display a variety of network-related details, such as open ports, current connections, and their associated IP addresses.

---

2. Syntax of `netstat` Command

General Syntax

bash

Copy code

```
netstat [options]
```

- [options]: Flags or parameters that modify the output of the command.

---

3. Common Options for `netstat` Command

a. `-a` - Display All Connections and Listening Ports

- Description: Shows all active connections and listening ports.

Example:
bash
Copy code
```
netstat -a
```

-

b. `-n` - Show Numerical Addresses

- Description: Displays addresses and port numbers in numerical format instead of resolving them to hostnames.

Example:
bash
Copy code
```
netstat -n
```

-

c. `-o` - Show PID (Process ID)

- Description: Displays the process ID (PID) associated with each connection or listening port.

Example:
bash
Copy code
```
netstat -o
```

-

d. `-p [protocol]` - Show Connections for Specific Protocol

- Description: Filters the output to show connections for a specific protocol (e.g., TCP or UDP).

Example:
bash
Copy code
```
netstat -p tcp
```

-

e. `-r` - Display Routing Table

- **Description: Displays the routing table of the system, showing the paths network traffic takes.**

Example:
bash
Copy code
```
netstat -r
```

-

f. `-s` - Display Network Statistics for Each Protocol

- **Description: Shows detailed statistics for each network protocol (TCP, UDP, ICMP, etc.).**

Example:
bash
Copy code
```
netstat -s
```

-

g. `-e` - Display Ethernet Statistics

- **Description: Displays Ethernet statistics, such as the number of bytes, packets, and errors.**

Example:
bash
Copy code
```
netstat -e
```

-

h. `-b` - Show Executable Involved in Creating Connections

- **Description: Displays the executable (program) associated with each connection or listening port.**

Example:
bash
Copy code
```
netstat -b
```

---

34. nslookup - Look up IP addresses and domain names.

1. Overview of the `nslookup` Command

What is the `nslookup` Command?

The `nslookup` (Name Server Lookup) command is used to query Domain Name System (DNS) to obtain information about domain names and IP addresses. It allows you to look up domain names to find corresponding IP addresses, and vice versa, and is a helpful tool for troubleshooting DNS-related issues.

The `nslookup` command is commonly used to:

- **Resolve domain names to IP addresses.**
- **Query DNS records (e.g., A records, MX records).**
- **Troubleshoot DNS server issues.**
- **Verify the availability of a domain name.**

---

2. Syntax of `nslookup` Command

General Syntax

bash

Copy code

```
nslookup [domain-name] [dns-server]
```

- **[domain-name]: The domain name you want to look up.**
- **[dns-server]: (Optional) The DNS server to use for the lookup. If not specified, the default DNS server of the system will be used.**

---

3. Common Options for `nslookup` Command

a. Without any Arguments

- Description: Running `nslookup` without any arguments enters interactive mode, allowing you to perform multiple queries.

Example:
bash
Copy code
```
nslookup
```

- This starts the interactive mode where you can enter domain names to query.

b. `-type=type` - Specify Query Type

- Description: Allows you to specify the type of DNS record you wish to look up (e.g., A, MX, CNAME).

Example:
bash
Copy code
```
nslookup -type=mx example.com
```

- This command queries the MX (Mail Exchange) records for `example.com`.

c. `-timeout=seconds` - Set Query Timeout

- Description: Specifies the timeout duration in seconds for the DNS query. By default, `nslookup` waits for 5 seconds before it times out.

Example:
bash
Copy code
```
nslookup -timeout=10 example.com
```

- This sets the timeout to 10 seconds for the query.

d. `-debug` - Show Debugging Information

- Description: Provides detailed debugging information about the DNS query process, including the query's result and the DNS server's response.

Example:
bash
Copy code
```
nslookup -debug example.com
```

-

e. `-port=port` - Specify Port Number

- **Description: Allows you to specify the port number to use for querying the DNS server. By default, DNS queries are sent over port 53.**

Example:
bash
Copy code

```
nslookup -port=5353 example.com
```

35. hostname - Display or set the computer's hostname.

The `hostname` command is used to display or set the system's hostname, which identifies the system on a network.

Basic Syntax

bash

Copy code

```
hostname [OPTION] [NEW_HOSTNAME]
```

1. Display the Current Hostname

bash

Copy code

```
hostname
```

- **Outputs the system's current hostname.**

2. Set a New Hostname (Requires Root Privileges)

bash

Copy code

```
sudo hostname new-hostname
```

- Temporarily sets the hostname until the next reboot.

3. Display Fully Qualified Domain Name (FQDN)

bash

Copy code

```
hostname -f
```

4. Display the Short Hostname

bash

Copy code

```
hostname -s
```

5. Display the Domain Name

bash

Copy code

```
hostname -d
```

6. Get the IP Address of the Hostname

bash

Copy code

```
hostname -i
```

7. Display Network Node (NIS/YP) Domain Name

bash

Copy code

```
hostname -y
```

8. Show All IP Addresses Associated with the Host

bash

Copy code

```
hostname -I
```

36. arp - Display and modify the ARP cache.

The `arp` (Address Resolution Protocol) command is used to view and manipulate the ARP table on a system. It is primarily used to map IP addresses to MAC (Media Access Control) addresses.

Basic Syntax

bash

Copy code

```
arp [OPTION] [hostname]
```

Usage

1. Display the ARP Table

bash

Copy code

```
arp -a
```

- **Lists all current ARP entries, showing IP addresses and their corresponding MAC addresses.**

2. Add a New ARP Entry

bash

Copy code

```
sudo arp -s <IP_ADDRESS> <MAC_ADDRESS>
```

- **Manually adds a static ARP entry to the table.**

3. Delete an ARP Entry

bash

Copy code

```
sudo arp -d <IP_ADDRESS>
```

- **Removes a specific entry for the given IP address.**

4. Show Specific ARP Entry

bash

Copy code

```
arp <hostname>
```

- Displays the ARP entry for the specified hostname or IP address.

Options

Option	Description
`-a [hostname]`	Display all ARP entries or a specific one if the hostname is provided.
`-d <IP_ADDRESS>`	Delete the ARP entry for the specified IP address.
`-s <IP> <MAC>`	Add a new static ARP entry for the given IP and MAC addresses.
`-v`	Enable verbose mode to show additional details.
`-n`	Display numerical addresses instead of resolving hostnames.

Practical Examples

1. View All ARP Entries

bash

Copy code

```
arp -a
```

- Shows the ARP cache with associated IP-MAC address mappings.

2. Add a Static ARP Entry

bash

Copy code

```
sudo arp -s 192.168.1.10 00:14:22:01:23:45
```

- Maps the IP `192.168.1.10` to the MAC address `00:14:22:01:23:45`.

3. Remove an ARP Entry

bash

Copy code

```
sudo arp -d 192.168.1.10
```

- Deletes the ARP cache entry for the IP address `192.168.1.10`.

4. Lookup a Specific Host

bash

Copy code

```
arp 192.168.1.1
```

- Displays the ARP information for the IP address `192.168.1.1`.

ARP Cache Management

View the ARP Cache in Linux

bash

Copy code

```
ip neigh show
```

- Modern systems use the `ip` command for viewing the ARP table.

Flush the Entire ARP Cache

bash

Copy code

```
sudo ip -s -s neigh flush all
```

List ARP Cache Entries in Windows

cmd

Copy code

```
arp -a
```

Delete ARP Entry in Windows

cmd

Copy code

```
arp -d <IP_ADDRESS>
```

37. route - Display or modify the routing table.

The `route` command is used to view and manipulate the IP routing table. It allows network administrators to define and manage static routes for a system to determine how packets are sent across a network.

Basic Syntax

bash

Copy code

```
route [OPTION] [COMMAND] [TARGET]
```

1. View the Current Routing Table

bash

Copy code

```
route
```

- **Displays the system's current routing table.**

2. View the Routing Table with Numeric Addresses

bash

Copy code

```
route -n
```

- **Displays the routing table without resolving IP addresses to hostnames.**

3. Add a Static Route for a Network

Syntax:

sudo route add -net <NETWORK> netmask <NETMASK> gw <GATEWAY>

bash

Copy code

```
sudo route add -net 192.168.2.0 netmask 255.255.255.0 gw 192.168.1.1
```

- Adds a static route to the `192.168.2.0/24` network via the gateway `192.168.1.1`.

4. Add a Static Route to a Host

Syntax:

sudo route add -host <IP_ADDRESS> gw <GATEWAY>

bash

Copy code

```
sudo route add -host 10.0.0.5 gw 192.168.1.1
```

- Adds a static route to the host `10.0.0.5` via the gateway `192.168.1.1`.

5. Remove a Network Route

Syntax:

sudo route del -net <NETWORK> netmask <NETMASK>

bash

Copy code

```
sudo route del -net 192.168.2.0 netmask 255.255.255.0
```

- Deletes the route to the `192.168.2.0/24` network.

6. Remove a Host Route

bash

Copy code

```
sudo route del -host 10.0.0.5
```

- Deletes the route to the host `10.0.0.5`.

7. Add a Default Gateway

bash

Copy code

```
sudo route add default gw 192.168.1.1
```

- Sets `192.168.1.1` as the default gateway for the system.

8. Delete the Default Gateway

bash

Copy code

```
sudo route del default
```

- Removes the default gateway.

38. telnet - Connect to remote hosts using Telnet.

1. Connect to a Remote Host on Port 23 (Default Telnet Port)

bash

Copy code

```
telnet 192.168.1.1
```

- Connects to the host with IP `192.168.1.1` on the default Telnet port.

2. Connect to a Remote Server on a Specific Port

bash

Copy code

```
telnet example.com 80
```

- Connects to `example.com` on port 80 to test HTTP services.

3. Send a Manual HTTP Request

bash

Copy code

```
telnet example.com 80
GET / HTTP/1.1
Host: example.com
```

- Sends an HTTP request to `example.com` and retrieves the response.

4. Enable Debugging Mode

bash

Copy code

```
telnet -d 192.168.1.1
```

- Enables debugging to show connection details while connecting to `192.168.1.1`.

5. Specify a Custom Escape Character

bash

Copy code

```
telnet -e ^C 192.168.1.1
```

- Sets the escape character to `Ctrl + C`.

6. Test an SMTP Server

bash

Copy code

```
telnet mail.example.com 25
```

- Connects to an SMTP server on port 25 to test mail server communication.

39. ftp - Transfer files to/from remote FTP servers.

The `ftp` command is used to transfer files between a local and a remote system over the File Transfer Protocol. It provides an interactive text interface to manage file uploads, downloads, and directory navigation on the remote server.

Basic Syntax

bash

Copy code

```
ftp [OPTIONS] [hostname or IP address]
```

Usage

1. Connect to an FTP Server

bash

Copy code

```
ftp <hostname_or_IP>
```

- **Connects to the specified FTP server.**

2. Authenticate on the Server

- **Once connected, provide the required username and password when prompted.**

3. Upload or Download Files

Use commands like `put` to upload files or `get` to download files during the session.

Options

Option	Description
`-v`	**Enables verbose mode, displaying detailed responses from the FTP server.**
`-n`	**Prevents automatic login upon connection.**
`-i`	**Disables interactive prompts for each file transfer.**

`-g`	Disables wildcard expansion in filenames.
`-d`	Enables debugging to show all FTP commands sent to the server.
`-p`	Enables passive mode for data transfer.

Common Commands Within an FTP Session

Command	Description
`open`	Connects to a different FTP server.
`close`	Disconnects from the current FTP server.
`quit`	Exits the FTP session.
`get`	Downloads a file from the remote server to the local machine.
`mget`	Downloads multiple files using wildcards (e.g., `mget *.txt`).
`put`	Uploads a file from the local machine to the remote server.
`mput`	Uploads multiple files using wildcards (e.g., `mput *.csv`).
`ls`	Lists files and directories on the remote server.

`dir`	Displays a detailed listing of files and directories.
`pwd`	Shows the current working directory on the remote server.
`cd`	Changes the directory on the remote server.
`lcd`	Changes the local working directory.
`delete`	Deletes a file on the remote server.
`mkdir`	Creates a directory on the remote server.
`rmdir`	Removes a directory on the remote server.
`ascii`	Switches to ASCII transfer mode (for text files).
`binary`	Switches to binary transfer mode (for binary files).
`status`	Displays the current FTP settings.

Practical Examples

1. Connect to an FTP Server

bash

Copy code

```
ftp ftp.example.com
```

- Connects to the server `ftp.example.com`.

2. Log in Without Automatic Authentication

bash

Copy code

```
ftp -n ftp.example.com
```

- Connects without auto-login; credentials must be entered manually.

3. Download a File

bash

Copy code

```
ftp> get example.txt
```

- Downloads the file `example.txt` from the remote server to the local system.

4. Upload a File

bash

Copy code

```
ftp> put localfile.txt
```

- Uploads the file `localfile.txt` from the local machine to the remote server.

5. Download Multiple Files

bash

Copy code

```
ftp> mget *.jpg
```

- Downloads all files with the `.jpg` extension.

6. Upload Multiple Files

bash

Copy code

```
ftp> mput *.csv
```

- Uploads all `.csv` files to the remote server.

7. Enable Binary Transfer Mode

bash

Copy code

```
ftp> binary
```

- Ensures files are transferred in binary mode to prevent corruption.

8. Change Directory on the Server

bash

Copy code

```
ftp> cd /path/to/directory
```

- Navigates to the specified directory on the remote server.

40. net - Manage network resources.

The `net` command is used in Windows to manage network resources, services, user accounts, and more. It provides a suite of subcommands to perform administrative tasks on local and remote computers.

1. `net accounts`

Description

Manages user account policies and password settings.

Syntax

bash

Copy code

```
net accounts [/minpwlen:length] [/maxpwage:days]
[/lockoutthreshold:number] [/domain]
```

Example

bash

Copy code

```
net accounts /minpwlen:8 /maxpwage:30
```

- **Sets the minimum password length to 8 characters and the maximum password age to 30 days.**

---

2. `net computer`

Description

Adds or removes computers from a domain.

Syntax

bash

Copy code

```
net computer \\<computername> {/add | /del}
```

Example

bash

Copy code

```
net computer \\Workstation01 /add
```

- **Adds `Workstation01` to the domain.**

---

3. `net config`

Description

Displays or modifies network service configurations.

Syntax

bash

Copy code

```
net config [server | workstation]
```

Example

bash

Copy code

```
net config workstation
```

- **Displays configuration details for the workstation service.**

---

4. `net continue`

Description

Resumes a paused service.

Syntax

bash

Copy code

```
net continue <service_name>
```

Example

bash

Copy code

```
net continue "Print Spooler"
```

- **Resumes the Print Spooler service.**

---

5. `net file`

Description

Lists or closes open files on a network.

Syntax

bash

Copy code

```
net file [ID] [/close]
```

Example

bash

Copy code

```
net file

net file 1234 /close
```

- Lists open files and closes the file with ID `1234`.

---

6. `net group`

Description

Manages global groups on a domain.

Syntax

bash

Copy code

```
net group <groupname> [/add | /delete] [/domain]
```

Example

bash

Copy code

```
net group "Sales Team" /add /domain
```

- Adds the group `Sales Team` to the domain.

---

7. `net help`

Description

Provides help for `net` commands.

Syntax

bash

Copy code

```
net help <command>
```

Example

bash

Copy code

```
net help user
```

- **Displays help for the `net user` command.**

---

8. `net localgroup`

Description

Manages local user groups.

Syntax

bash

Copy code

```
net localgroup <groupname> [/add | /delete] [/domain]
```

Example

bash

Copy code

```
net localgroup "Administrators" JohnDoe /add
```

- Adds `JohnDoe` to the `Administrators` group.

---

9. `net name`

Description

Manages messaging names.

Syntax

bash

Copy code

```
net name [name] [/add | /delete]
```

Example

bash

Copy code

```
net name Support /add
```

- Adds the messaging name `Support`.

---

10. `net pause`

Description

Pauses a service.

Syntax

bash

Copy code

```
net pause <service_name>
```

Example

bash

Copy code

```
net pause "Print Spooler"
```

- **Pauses the Print Spooler service.**

---

11. `net print`

Description

Displays or controls print jobs.

Syntax

bash

Copy code

```
net print \\<computername>\<sharename>
```

Example

bash

Copy code

```
net print \\Server01\Printer1
```

- Displays print jobs for the printer `Printer1` on `Server01`.

---

12. `net send` (Deprecated)

Description

Sends messages to other users or computers.

Syntax

bash

Copy code

```
net send <name> <message>
```

Example

bash

Copy code

```
net send JohnDoe "Meeting at 3 PM"
```

- Sends the message to `JohnDoe`.

---

13. `net session`

Description

Lists or ends active sessions.

Syntax

bash

Copy code

```
net session [\\<computername>] [/delete]
```

Example

bash

Copy code

```
net session

net session \\Workstation02 /delete
```

- **Lists active sessions and ends the session for `Workstation02`.**

---

14. `net share`

Description

Manages shared resources.

Syntax

bash

Copy code

```
net share [sharename] [=path] [/delete]
```

Example

bash

Copy code

```
net share PublicShare=C:\Public

net share PublicShare /delete
```

- Shares the folder `C:\Public` and deletes the share.

---

15. `net start`

Description

Starts a service.

Syntax

bash

Copy code

```
net start <service_name>
```

Example

bash

Copy code

```
net start "Print Spooler"
```

- Starts the Print Spooler service.

---

16. `net statistics`

Description

Displays network statistics for workstation or server.

Syntax

bash

Copy code

```
net statistics [workstation | server]
```

Example

bash

Copy code

```
net statistics workstation
```

- **Displays statistics for the workstation service.**

---

17. `net stop`

Description

Stops a service.

Syntax

bash

Copy code

```
net stop <service_name>
```

Example

bash

Copy code

```
net stop "Print Spooler"
```

- **Stops the Print Spooler service.**

---

18. `net time`

Description

Synchronizes or displays the network time.

Syntax

bash

Copy code

```
net time [\\<computername>] [/set] [/yes]
```

Example

bash

Copy code

```
net time \\Server01 /set /yes
```

- **Synchronizes the system time with `Server01`.**

---

19. `net use`

Description

Manages connections to network resources.

Syntax

bash

Copy code

```
net use [drive:] \\<computername>\<sharename> [/delete]
```

Example

bash

Copy code

```
net use Z: \\Server01\SharedFolder

net use Z: /delete
```

- Maps the shared folder to drive `Z:` and deletes the mapping.

---

20. `net user`

Description

Manages user accounts.

Syntax

bash

Copy code

```
net user <username> [password | *] [/add | /delete]
```

Example

bash

Copy code

```
net user JohnDoe password123 /add

net user JohnDoe /delete
```

- **Adds and deletes the user `JohnDoe`.**

---

21. `net view`

Description

Lists shared resources or computers.

Syntax

bash

Copy code

```
net view [\\<computername>]
```

Example

bash

Copy code

```
net view

net view \\Server01
```

- **Displays shared resources or computers.**

41. netsh - Network Shell for configuring network-related settings.

`netsh` **(Network Shell) is a powerful command-line utility for managing and configuring network-related settings on a Windows machine. It supports various contexts like interface, firewall, IP, routing, and more.**

1. General Commands

These commands are used to navigate or reset `netsh` **configurations.**

Syntax and Commands

bash

Copy code

```
netsh
```

```
netsh show commands
```

```
netsh <context> show help
```

```
netsh -c <context>
```

```
netsh reset
```

Examples

Enter `netsh` interactive mode:
bash
Copy code

```
netsh
```

-

Show available commands in a context:
bash
Copy code

```
netsh interface show commands
```

-

Reset network settings:
bash
Copy code

```
netsh int ip reset
```

-

---

2. Interface Context

Description

Manages network interfaces, including enabling/disabling interfaces, configuring IP addresses, and viewing interface details.

Syntax

bash

Copy code

```
netsh interface show interface

netsh interface set interface name="<InterfaceName>" admin=enabled

netsh interface ip set address name="<InterfaceName>" source=dhcp

netsh interface ip set address name="<InterfaceName>" source=static
addr=<IP> mask=<SubnetMask> gateway=<Gateway>
```

Examples

List all interfaces:
bash
Copy code

```
netsh interface show interface
```

-

Enable a specific interface:
bash
Copy code

```
netsh interface set interface name="Ethernet" admin=enabled
```

-

Set a static IP:
bash
Copy code

```
netsh interface ip set address name="Ethernet" source=static
addr=192.168.1.10 mask=255.255.255.0 gateway=192.168.1.1
```

-

Set IP to DHCP:
bash

Copy code

```
netsh interface ip set address name="Ethernet" source=dhcp
```

-

---

3. Firewall Context

Description

Manages Windows Firewall rules and configurations.

Syntax

bash

Copy code

```
netsh advfirewall show allprofiles

netsh advfirewall set allprofiles state on

netsh advfirewall firewall add rule name="<RuleName>" dir=in
action=allow protocol=TCP localport=<Port>

netsh advfirewall firewall delete rule name="<RuleName>"
```

Examples

Enable Windows Firewall:
bash
Copy code

```
netsh advfirewall set allprofiles state on
```

-

Add a firewall rule:
bash
Copy code

```
netsh advfirewall firewall add rule name="Allow_HTTP" dir=in
action=allow protocol=TCP localport=80
```

-

Delete a firewall rule:
bash
Copy code

```
netsh advfirewall firewall delete rule name="Allow_HTTP"
```

-

---

4. WLAN Context

Description

Manages wireless network settings, including profiles, connections, and configurations.

Syntax

bash

Copy code

```
netsh wlan show profiles

netsh wlan connect name="<ProfileName>"

netsh wlan set hostednetwork mode=allow ssid=<SSID> key=<Password>

netsh wlan start hostednetwork

netsh wlan stop hostednetwork
```

Examples

Show all saved Wi-Fi profiles:
bash
Copy code

```
netsh wlan show profiles
```

-

Connect to a specific Wi-Fi network:
bash
Copy code

```
netsh wlan connect name="HomeWiFi"
```

-

Create a Wi-Fi hotspot:
bash
Copy code

```
netsh wlan set hostednetwork mode=allow ssid=MyHotspot key=Password123

netsh wlan start hostednetwork
```

-

---

5. IPSec Context

Description

Configures IP Security policies for network traffic.

Syntax

bash

Copy code

```
netsh ipsec static show policy

netsh ipsec static add policy name="<PolicyName>"
description="<Description>"

netsh ipsec static delete policy name="<PolicyName>"
```

Examples

Show all IPSec policies:
bash
Copy code

```
netsh ipsec static show policy
```

-

Add a new IPSec policy:
bash
Copy code

```
netsh ipsec static add policy name="SecurePolicy" description="Secure
Traffic Policy"
```

-

Delete an IPSec policy:
bash
Copy code

```
netsh ipsec static delete policy name="SecurePolicy"
```

-

---

6. Routing Context

Description

Configures static routes and routing-related settings.

Syntax

bash

Copy code

```
netsh routing ip show config

netsh routing ip add persistentroute destination=<IP>
mask=<SubnetMask> gateway=<Gateway> metric=<Metric>

netsh routing ip delete persistentroute destination=<IP>
```

Examples

Show routing configuration:
bash
Copy code

```
netsh routing ip show config
```

-

Add a static route:
bash

Copy code

```
netsh routing ip add persistentroute destination=192.168.2.0
mask=255.255.255.0 gateway=192.168.1.1 metric=1
```

-

Delete a static route:
bash
Copy code

```
netsh routing ip delete persistentroute destination=192.168.2.0
```

-

---

7. Diagnostics Context

Description

Performs network diagnostics and troubleshooting.

Syntax

bash

Copy code

```
netsh diag show test

netsh diag ping ipaddress=<IP>

netsh diag connect ipaddress=<IP>
```

Examples

Run a full diagnostics test:
bash
Copy code

```
netsh diag show test
```

-

Ping a specific IP:
bash

Copy code

```
netsh diag ping ipaddress=8.8.8.8
```

-

---

nbtstat - Display statistics and current connections using NetBIOS over TCP/IP.

`nbtstat` is a Windows command-line utility used to display and troubleshoot NetBIOS over TCP/IP (NetBT) statistics and connections. It provides information about NetBIOS names, sessions, and caches.

1. General Syntax

bash

Copy code

```
nbtstat [options]
```

Where `[options]` are specific flags or parameters used to perform tasks like listing the name table, cache, or active sessions.

---

2. Commands and Descriptions

Command	Description
`nbtstat -a <hostname>`	**Displays the NetBIOS name table of a remote machine, specified by `<hostname>`.**
`nbtstat -A <IP>`	**Displays the NetBIOS name table of a remote machine using its IP address.**

`nbtstat -c`	Displays the contents of the NetBIOS name cache, which maps NetBIOS names to IP addresses.
`nbtstat -n`	Lists local NetBIOS names that have been registered on the machine.
`nbtstat -r`	Displays the count of NetBIOS names resolved by broadcast and via WINS.
`nbtstat -R`	Purges and reloads the NetBIOS name cache.
`nbtstat -s`	Displays NetBIOS sessions and their statuses for the client.
`nbtstat -S`	Displays the current NetBIOS sessions and their status with IP addresses.
`nbtstat -RR`	Sends a name release request to WINS servers and re-registers the names.

---

3. Examples

3.1 View the NetBIOS Name Table of a Remote Machine by Hostname

Command:

bash

Copy code

```
nbtstat -a <hostname>
```

Example:

bash

Copy code

```
nbtstat -a RemotePC
```

Description:
Displays the NetBIOS name table for the machine `RemotePC`.

3.2 View the NetBIOS Name Table of a Remote Machine by IP Address

Command:

bash

Copy code

```
nbtstat -A <IP>
```

Example:

bash

Copy code

```
nbtstat -A 192.168.1.10
```

Description:
Shows the NetBIOS name table for the machine at IP address `192.168.1.10`.

3.3 Display the NetBIOS Name Cache

Command:

bash

Copy code

```
nbtstat -c
```

Example:

bash

Copy code

```
nbtstat -c
```

Description:
Lists all entries in the local NetBIOS name cache.

3.4 List Local NetBIOS Names

Command:

bash

Copy code

```
nbtstat -n
```

Example:

bash

Copy code

```
nbtstat -n
```

Description:
Displays local NetBIOS names registered by the machine.

3.5 Show Name Resolution Statistics

Command:

bash

Copy code

```
nbtstat -r
```

Example:

bash

Copy code

```
nbtstat -r
```

Description:
Shows the count of NetBIOS names resolved by broadcast or via WINS.

3.6 Purge and Reload the NetBIOS Name Cache

Command:

bash

Copy code

```
nbtstat -R
```

Example:

bash

Copy code

```
nbtstat -R
```

Description:
Clears the NetBIOS name cache and reloads it.

3.7 Display Current NetBIOS Sessions (with Hostnames)

Command:

bash

Copy code

```
nbtstat -s
```

Example:

bash

Copy code

```
nbtstat -s
```

Description:
Shows all active NetBIOS sessions with hostnames.

3.8 Display Current NetBIOS Sessions (with IP Addresses)

Command:

bash

Copy code

```
nbtstat -S
```

Example:

bash

Copy code

```
nbtstat -S
```

Description:
Lists all active NetBIOS sessions and displays their IP addresses.

3.9 Release and Re-register NetBIOS Names

Command:

bash

Copy code

```
nbtstat -RR
```

Example:

bash

Copy code

```
nbtstat -RR
```

Description:
Sends a name release request to WINS servers and re-registers all NetBIOS names

55. systeminfo - Display detailed system information.

`systeminfo` **is a command-line tool that provides detailed information about the computer system's configuration, including OS version, memory, disk space, network adapters, and more.**

1. General Syntax

bash

Copy code

```
systeminfo [options]
```

Where `[options]` are flags or parameters used to modify or customize the output.

---

2. Commands and Options

Command/Option	Description
`systeminfo`	Displays detailed system information about the local computer.
`/s <hostname>`	Specifies a remote computer (hostname or IP address) to retrieve system information.
`/u <username>`	Runs the command with the specified username (for remote machines).
`/p <password>`	Specifies the password for the username (for remote machines).
`/fi <filter>`	Filters the output based on specified criteria (e.g., OS, memory).
`/fo <format>`	Specifies the output format (`TABLE`, `LIST`, `CSV`).
`/nh`	No header, used with `/fo` to exclude column headers in the output.
`/?`	Displays help information for the command.

---

3. Examples

3.1 Display Basic System Information

Command:

bash

Copy code

```
systeminfo
```

Example:

bash

Copy code

```
systeminfo
```

Description:
Displays detailed information about the local computer system, including OS version, manufacturer, memory, and more.

3.2 Display System Information for a Remote Computer

Command:

bash

Copy code

```
systeminfo /s <hostname>
```

Example:

bash

Copy code

```
systeminfo /s 192.168.1.10
```

Description:
Fetches and displays system information for the remote machine with the IP address `192.168.1.10`.

3.3 Display System Information with Specific User Credentials

Command:

bash

Copy code

```
systeminfo /s <hostname> /u <username> /p <password>
```

Example:

bash

Copy code

```
systeminfo /s 192.168.1.10 /u Administrator /p P@ssw0rd
```

Description:
Fetches system information for a remote computer and runs the command using the provided username and password.

3.4 Filter Output Based on a Specific Criterion

Command:

bash

Copy code

```
systeminfo /fi "<filter>"
```

Example:

bash

Copy code

```
systeminfo /fi "OS"
```

Description:
Filters and displays only the information related to the operating system. Other filters can be used, like "Memory" or "Host Name."

3.5 Display Output in a Specific Format

Command:

bash

Copy code

```
systeminfo /fo <format>
```

Example:

bash

Copy code

```
systeminfo /fo list
```

Description:
Displays the output in a list format. Other available formats include `TABLE` and `CSV`.

3.6 Exclude Headers in the Output

Command:

bash

Copy code

```
systeminfo /fo <format> /nh
```

Example:

bash

Copy code

```
systeminfo /fo csv /nh
```

Description:
Displays the output in CSV format but without the header row.

3.7 Display Help Information

Command:

bash

Copy code

```
systeminfo /?
```

Example:

bash

Copy code

```
systeminfo /?
```

Description:
Displays help information about the `systeminfo` command and its options.

56. tasklist - List running processes.

`tasklist` provides a snapshot of all running processes on a machine, which can be useful for system monitoring, troubleshooting, or process management. It allows you to filter the list of tasks based on various criteria, such as memory usage or process name.

---

1. General Syntax

bash

Copy code

```
tasklist [options]
```

Where `[options]` are flags or parameters used to modify or filter the output of the command.

---

2. Commands and Options

Command/Option	Description
`tasklist`	Displays a list of currently running processes on the local computer.
`/s <hostname>`	Specifies the name or IP address of a remote machine to query tasks.
`/u <username>`	Runs the command using the specified username (for remote machines).
`/p <password>`	Specifies the password for the username (for remote machines).
`/m <module>`	Lists all tasks that are using a specific module or DLL.

`/v`	Displays verbose information about each process (including memory usage).
`/fi <filter>`	Filters the output based on specified criteria (e.g., process name, PID).
`/fo <format>`	Specifies the output format (`TABLE`, `LIST`, `CSV`).
`/nh`	Excludes the column headers in the output (used with `/fo`).
`/t`	Terminates the specified process along with any child processes.
`/?`	Displays help information for the command.

---

3. Examples

3.1 Display All Running Processes

Command:

bash

Copy code

```
tasklist
```

Example:

bash

Copy code

```
tasklist
```

Description:
Displays a list of all currently running processes on the local computer.

3.2 Display All Running Processes on a Remote Computer

Command:

bash

Copy code

```
tasklist /s <hostname>
```

Example:

bash

Copy code

```
tasklist /s 192.168.1.10
```

Description:
Displays a list of running processes on a remote machine at IP address `192.168.1.10`.

3.3 Display All Running Processes for a Specific User

Command:

bash

Copy code

```
tasklist /u <username> /p <password>
```

Example:

bash

Copy code

```
tasklist /u Administrator /p P@ssw0rd
```

Description:
Displays all running processes on the local computer or a remote machine while using the provided username and password.

3.4 Display Processes Using a Specific Module

Command:

bash

Copy code

```
tasklist /m <module>
```

Example:

bash

Copy code

```
tasklist /m kernel32.dll
```

Description:
Displays all tasks that are using the specified module (in this case, `kernel32.dll`).

3.5 Display Detailed Process Information (Verbose Output)

Command:

bash

Copy code

```
tasklist /v
```

Example:

bash

Copy code

```
tasklist /v
```

Description:
Displays detailed information about each running process, including memory usage and additional process details.

3.6 Filter the Process List by Name

Command:

bash

Copy code

```
tasklist /fi "imagename eq <process_name>"
```

Example:

bash

Copy code

```
tasklist /fi "imagename eq chrome.exe"
```

Description:
Filters the process list to display only processes named `chrome.exe`.

3.7 Filter the Process List by PID

Command:

bash

Copy code

```
tasklist /fi "pid eq <PID>"
```

Example:

bash

Copy code

```
tasklist /fi "pid eq 1234"
```

Description:
Displays information about the process with the specified Process ID (PID).

3.8 Display Output in a Specific Format

Command:

bash

Copy code

```
tasklist /fo <format>
```

Example:

bash

Copy code

```
tasklist /fo list
```

Description:
Displays the output in a list format. Other available formats include `TABLE` and `CSV`.

3.9 Exclude Headers from Output

Command:

bash

Copy code

```
tasklist /fo <format> /nh
```

Example:

bash

Copy code

```
tasklist /fo csv /nh
```

Description:
Displays the output in CSV format, excluding the header row.

3.10 Terminate a Process

Command:

bash

Copy code

```
tasklist /t /fi "imagename eq <process_name>"
```

Example:

bash

Copy code

```
tasklist /t /fi "imagename eq notepad.exe"
```

Description:
Terminates the process `notepad.exe` and any associated child processes.

3.11 Display Help Information

Command:

bash

Copy code

```
tasklist /?
```

Example:

bash

Copy code

```
tasklist /?
```

Description:
Displays help information about the `tasklist` command and its available options.

57. taskkill - Terminate processes or applications.

The `taskkill` command in Windows is used to terminate processes by their process ID (PID) or image name. This command allows you to stop applications and processes from running on your system.

Basic Syntax

bash

Copy code

```
taskkill [/f] [/im <imagename>] [/pid <pid>] [/t] [/fi <filter>]
```

- `/f`: Forces the process to terminate.
- `/im <imagename>`: Specifies the image name of the process to be terminated (e.g., `notepad.exe`).
- `/pid <pid>`: Specifies the process ID to be terminated.
- `/t`: Terminates the specified process and all its child processes.
- `/fi <filter>`: Specifies a filter to select a set of tasks, such as processes running as a certain user.

Usage

1. Terminate a Process by Image Name

bash

Copy code

```
taskkill /im notepad.exe
```

- Terminates all processes named `notepad.exe`.

2. Forcefully Terminate a Process

bash

Copy code

```
taskkill /f /im notepad.exe
```

- Forcefully terminates `notepad.exe` without prompting for confirmation.

3. Terminate a Process by PID (Process ID)

bash

Copy code

```
taskkill /pid 1234
```

- Terminates the process with PID `1234`.

4. Terminate a Process and Its Child Processes

bash

Copy code

```
taskkill /im notepad.exe /t
```

- Terminates `notepad.exe` along with any child processes spawned by it.

5. Terminate Multiple Processes

bash

Copy code

```
taskkill /im notepad.exe /im calc.exe
```

- Terminates both `notepad.exe` and `calc.exe` processes.

Options

Option	Description
`/f`	Forcefully terminates the process, ignoring any warning or prompt.
`/im <imagename>`	Specifies the name of the executable (image) of the process to terminate.
`/pid <pid>`	Specifies the PID of the process to terminate.
`/t`	Terminates the specified process and all of its child processes.
`/fi <filter>`	Allows you to filter tasks based on specific criteria (e.g., `status eq running`).
`/s <computer>`	Specifies the remote computer to execute the command on.
`/u <username>`	Specifies the username to use for authentication on the remote computer.
`/p <password>`	Specifies the password to use for authentication.

Practical Examples

1. Terminate a Process by Image Name

bash

Copy code

```
taskkill /im firefox.exe
```

- Terminates all instances of `firefox.exe`.

2. Forcefully Terminate a Process

bash

Copy code

```
taskkill /f /im chrome.exe
```

- Forcefully terminates `chrome.exe` without asking for confirmation.

3. Terminate a Process by PID

bash

Copy code

```
taskkill /pid 4567
```

- Terminates the process with PID `4567`.

4. Terminate Multiple Processes

bash

Copy code

```
taskkill /f /im notepad.exe /im mspmsnsv.exe
```

- Forcefully terminates both `notepad.exe` and `mspmsnsv.exe`.

5. Terminate a Process and Its Child Processes

bash

Copy code

```
taskkill /im explorer.exe /t
```

- Terminates `explorer.exe` and any processes launched by it, including Windows Explorer.

6. Use a Filter to Terminate a Process

bash

Copy code

```
taskkill /fi "status eq running" /f
```

- Forcefully terminates all processes with a running status.

7. Terminate Processes on a Remote Computer

bash

Copy code

```
taskkill /s remotePC /u user /p password /im notepad.exe
```

- Terminates the `notepad.exe` process on the remote computer `remotePC` using the specified user credentials.

Filter Options

The `/fi` option allows you to specify various filters to narrow down the list of processes. Some common filters include:

Filter	Description

`status eq running`	Filters processes that are currently running.
`status eq stopped`	Filters processes that are currently stopped.
`imagename eq <name>`	Filters processes by their image name (e.g., `imagename eq chrome.exe`).
`pid eq <pid>`	Filters processes by their PID.
`username eq <name>`	Filters processes by the user who is running them.

Example: Use Filter to Terminate a Process

bash

Copy code

```
taskkill /fi "imagename eq notepad.exe"
```

- Terminates all processes named `notepad.exe`.

58. msconfig - System Configuration Utility.

`msconfig` is a tool that helps you configure the startup environment of the system. It allows users to modify startup programs, system services, boot options, and more. It is often used to troubleshoot performance issues or diagnose system problems by controlling which programs and services are loaded during startup.

1. General Syntax

bash

Copy code

```
msconfig
```

This command will open the System Configuration Utility window.

---

2. Key Sections in `msconfig`

When you launch `msconfig`, you will find several tabs to configure the system:

1. General Tab: Choose the type of startup you want (Normal, Diagnostic, or Selective startup).
2. Boot Tab: Configure boot settings, including advanced options and Safe Mode.
3. Services Tab: Manage services that start with Windows (you can disable or enable specific services).
4. Startup Tab: Manage programs that start automatically when Windows starts (in Windows 7 and later, this is replaced by the Task Manager).
5. Tools Tab: Provides shortcuts to various system tools for advanced configuration.

---

3. Commands and Options

The `msconfig` utility itself does not have specific command-line arguments, but it has several configurable options once opened through the utility's interface.

Command/Option	Description
`msconfig`	Opens the System Configuration Utility window.
`/selective`	Opens `msconfig` in Selective Startup mode, allowing you to choose which services and programs to disable.
`/diagnostic`	Opens `msconfig` in Diagnostic Startup mode, loading only essential services and drivers.

`/boot`	Opens the Boot tab for configuring boot options, like Safe Mode.
`/services`	Opens the Services tab for managing services.
`/startup`	Opens the Startup tab for managing startup programs.
`/tools`	Opens the Tools tab for accessing various system tools.

---

4. Examples

4.1 Open `msconfig` in Normal Startup Mode

Command:

bash

Copy code

```
msconfig
```

Example:

bash

Copy code

```
msconfig
```

Description:
This command opens the System Configuration Utility window in Normal startup mode, where you can configure startup programs, services, and boot options.

4.2 Open `msconfig` in Selective Startup Mode

Command:

bash

Copy code

```
msconfig /selective
```

Example:

bash

Copy code

```
msconfig /selective
```

Description:
This opens the System Configuration Utility in Selective Startup mode, where you can manually choose which services and startup programs to enable or disable.

4.3 Open `msconfig` in Diagnostic Startup Mode

Command:

bash

Copy code

```
msconfig /diagnostic
```

Example:

bash

Copy code

```
msconfig /diagnostic
```

Description:
This opens the System Configuration Utility in Diagnostic Startup mode, which only loads essential system services and drivers, making it useful for troubleshooting.

4.4 Access Boot Configuration Settings

Command:

bash

Copy code

```
msconfig /boot
```

Example:

bash

Copy code

```
msconfig /boot
```

Description:
This opens the Boot tab where you can configure boot options like Safe Mode, boot logging, and other advanced boot settings.

4.5 Access System Services Settings

Command:

bash

Copy code

```
msconfig /services
```

Example:

bash

Copy code

```
msconfig /services
```

Description:
This opens the Services tab where you can manage which services run at startup. You can disable or enable specific services to troubleshoot or optimize system performance.

4.6 Access Startup Configuration

Command:

bash

Copy code

```
msconfig /startup
```

Example:

bash

Copy code

```
msconfig /startup
```

Description:
This opens the Startup tab (replaced by Task Manager in Windows 8 and later), where you can manage startup programs and prevent unnecessary programs from running at startup.

4.7 Access System Tools

Command:

bash

Copy code

```
msconfig /tools
```

Example:

bash

Copy code

```
msconfig /tools
```

Description:
This opens the Tools tab in msconfig, providing shortcuts to various system tools like Event Viewer, System Restore, Command Prompt, etc.

59. regedit - Registry Editor.

The `regedit` tool in Windows allows users to view and modify the Windows registry, which can significantly impact system behavior. Below is a comprehensive guide of commands and options related to `regedit`, including how to gain full access and perform various operations.

Note: Modifying the registry can be risky. Always back up your registry before making changes to prevent system instability.

Overview of `regedit` Commands

General Syntax:

bash

Copy code

```
regedit [options]
```

The `regedit` command is primarily used for accessing, exporting, importing, and modifying the Windows registry.

Key Options and Commands

1. Open the Registry Editor GUI:

This is the most common usage of `regedit` to launch the Windows Registry Editor.

bash

Copy code

```
regedit
```

- **Opens the GUI where you can browse and modify registry keys and values.**

---

2. Export Registry Keys to a `.reg` File:

To back up the registry or specific keys:

bash

Copy code

```
regedit /e <filename> <key_path>
```

Example:
bash
Copy code

```
regedit /e "C:\Backup\myregistry.reg"
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft"
```

-
    - **Exports the `Microsoft` registry key under `HKEY_LOCAL_MACHINE` to a file.**

Additional Export Options:

- `/s` - Silent mode, no confirmation.
- `/f` - Forces the export even if the file already exists.

---

3. Import Registry File:

Import a `.reg` file back into the registry. This is typically used for restoring settings or applying registry tweaks:

bash

Copy code

```
regedit /s <filename>
```

Example:
bash
Copy code

```
regedit /s "C:\Backup\myregistry.reg"
```

-
    - **Imports the `myregistry.reg` file silently without any prompts.**

Additional Import Options:

- `/f` - Forces import, overwriting any conflicting registry keys.

---

4. Delete Registry Keys or Values:

To delete a registry key or value:

bash

Copy code

```
regedit /d <key_path>
```

Example:
bash
Copy code

```
regedit /d "HKEY_CURRENT_USER\Software\MyApp"
```

-
    - **Deletes the `MyApp` registry key under `HKEY_CURRENT_USER`.**

Important Notes:

- **Be cautious when deleting registry keys. Deleting critical keys can cause system issues.**
- **Use `regedit` to perform manual clean-ups of orphaned registry entries.**

---

5. Silent Export:

Export the registry or specific keys silently (no UI confirmation).

bash

Copy code

```
regedit /e <filename> /s
```

Example:
bash
Copy code

```
regedit /e "C:\Backup\system.reg" /s
```

-
    - **Exports the registry silently to `system.reg`.**

---

6. Silent Import:

Import registry settings without any user confirmation (silent mode).

bash

Copy code

```
regedit /s <filename>
```

Example:
bash
Copy code

```
regedit /s "C:\Backup\system.reg"
```

-
    - Imports the registry settings from `system.reg` without user interaction.

---

7. Force Operation:

Use `/f` to force operations like export or import, even if the file already exists.

bash

Copy code

```
regedit /e "C:\Backup\myregistry.reg" /f
```

Example:
bash
Copy code

```
regedit /e "C:\Backup\myregistry.reg" /f
```

-
    - Forces the export even if the file `myregistry.reg` already exists.

---

8. Open a Specific Registry Hive:

Use `regedit` to open a specific registry hive (e.g., `HKEY_LOCAL_MACHINE`) directly.

bash

Copy code

```
regedit /s "HKEY_LOCAL_MACHINE\Software"
```

Example:
bash
Copy code

```
regedit /s "HKEY_LOCAL_MACHINE\Software"
```

-
    - **This command opens the `Software` registry key under `HKEY_LOCAL_MACHINE` without launching the GUI.**

---

9. Open the Registry Editor with Full Access:

You can open the Registry Editor with full administrative privileges using the following steps:

- **Open Command Prompt as Administrator (right-click > Run as Administrator).**
- **Then enter:**

bash

Copy code

```
regedit
```

- **This ensures that the registry editor opens with full access to modify all registry keys and values.**

---

10. Access Registry via Command Prompt with Full Administrative Privileges:

You can also access the Registry Editor through Command Prompt with administrative access:

1. **Press `Windows + R` to open the Run dialog.**
2. **Type `cmd` and press Ctrl + Shift + Enter to run Command Prompt as administrator.**

In the Command Prompt window, type:
bash
Copy code

```
regedit
```

3.
    - This gives you full access to edit any registry key, as you are operating in Administrator mode.

60. eventvwr - Event Viewer.

Description: The `wevtutil` utility is used to manage event logs and event log subscriptions. It provides a set of commands that allow for querying, exporting, and clearing event logs, among other tasks.

Syntax:

bash

Copy code

```
wevtutil <subcommand> [options]
```

Common `wevtutil` Subcommands and Options

1. qe (Query Event Logs)

This subcommand is used to query event logs based on specific criteria.

Syntax:

bash

Copy code

```
wevtutil qe <LogName> [/f:<Format>] [/c:<NumberOfEvents>] [/rd:<Days>] [/k:<Key>]
```

Options:

- `<LogName>`: Name of the log (e.g., `System`, `Application`).
- `/f:<Format>`: Specifies the output format. Options are `text` or `xml`.

- `/c:<NumberOfEvents>`: Specifies the number of events to return.
- `/rd:<Days>`: Filters events generated within the last specified number of days.
- `/k:<Key>`: Filters events containing the specified keyword.

Example:

Query the last 5 events from the System log in text format:

bash

Copy code

```
wevtutil qe System /f:text /c:5
```

---

2. `epl` (Export Event Logs)

This subcommand is used to export event logs to a file in XML format.

Syntax:

bash

Copy code

```
wevtutil epl <LogName> <FileName>
```

Options:

- `<LogName>`: Name of the log (e.g., `Application`, `Security`).
- `<FileName>`: Path and filename for the exported log (e.g., `C:\Logs\log.xml`).

Example:

Export the Application log to a file named `app_log.xml`:

bash

Copy code

```
wevtutil epl Application C:\Logs\app_log.xml
```

---

3. `el` (List Event Logs)

This subcommand lists all available event logs on the system.

Syntax:

bash

Copy code

```
wevtutil el
```

Example:

List all event logs on the system:

bash

Copy code

```
wevtutil el
```

---

4. `gl` (Get Log Information)

This subcommand provides detailed information about a specified event log.

Syntax:

bash

Copy code

```
wevtutil gl <LogName>
```

Options:

- `<LogName>`: Name of the log (e.g., `System`, `Security`).

Example:

Get detailed information about the Security log:

bash

Copy code

```
wevtutil gl Security
```

---

5. `cl` (Clear Event Logs)

This subcommand is used to clear an event log.

Syntax:

bash

Copy code

```
wevtutil cl <LogName>
```

Options:

- `<LogName>`: Name of the log (e.g., `Application`, `System`).

Example:

Clear the System log:

bash

Copy code

```
wevtutil cl System
```

---

6. `sl` (Set Log Properties)

This subcommand is used to set properties on event logs, such as maximum size and retention policy.

Syntax:

bash

Copy code

```
wevtutil sl <LogName> /ms:<MaxSize> [/rt:<RetentionPolicy>]
```

Options:

- `<LogName>`: **Name of the log (e.g.,** `System`, `Application`**).**
- `/ms:<MaxSize>`: **Specifies the maximum size (in bytes) of the log file.**
- `/rt:<RetentionPolicy>`: **Specifies the retention policy (e.g.,** `AutoBackup`**).**

Example:

Set the maximum size of the Application log to 20 MB:

bash

Copy code

```
wevtutil sl Application /ms:20971520
```

---

7. `sd` (Set Security Descriptor)

This subcommand is used to set security descriptors for event logs.

Syntax:

bash

Copy code

```
wevtutil sd <LogName> <SecurityDescriptor>
```

Options:

- `<LogName>`: Name of the log (e.g., `System`, `Security`).
- `<SecurityDescriptor>`: Security descriptor in the standard SDDL (Security Descriptor Definition Language) format.

Example:

Set the security descriptor for the Application log:

bash

Copy code

```
wevtutil sd Application D:(A;;0x1f01ff;;;SY)(A;;0x1f01ff;;;BA)
```

---

8. gp (Get Publisher Information)

This subcommand retrieves information about event log publishers, which are responsible for writing events to the logs.

Syntax:

bash

Copy code

```
wevtutil gp <PublisherName>
```

Example:

Get information about a specific event log publisher:

bash

Copy code

```
wevtutil gp Microsoft-Windows-Security-Auditing
```

Additional `wevtutil` Usage Examples

Exporting System Log to an XML File:

bash

Copy code

```
wevtutil epl System C:\Logs\system_log.xml
```

Query the Last 10 Events from Application Log:

bash

Copy code

```
wevtutil qe Application /f:text /c:10
```

Clear the Security Log:

bash

Copy code

```
wevtutil cl Security
```

List All Available Logs:

bash

Copy code

```
wevtutil el
```

61. services.msc - Services management console.

`services.msc` is a Microsoft Management Console (MMC) snap-in that provides a graphical interface for managing Windows services. While there are no direct command-line commands for `services.msc` itself, you can manage the services using several actions within the **Services** window, and also via command-line tools like **`net`**, **`sc`**, and **PowerShell**.

Here's an overview of the actions available within `services.msc`, including both GUI actions and command-line equivalents:

How to Access `services.msc`

To open the **Services** window (via `services.msc`):

1. Press `Win + R` to open the **Run** dialog.
2. Type `services.msc` and press **Enter**.

Or, search for **`services.msc`** in the Windows search bar and click on the result.

Actions Available in `services.msc`

1. Start a Service

You can **start** a service that is currently stopped:

- **Right-click** on the service and select **Start**.

Command-Line Equivalent:

bash

Copy code

```
net start <ServiceName>
```

Or using **sc**:

bash

Copy code

```
sc start <ServiceName>
```

2. Stop a Service

You can **stop** a running service:

- **Right-click** on the service and select **Stop**.

Command-Line Equivalent:

bash

Copy code

```
net stop <ServiceName>
```

Or using **sc**:

bash

Copy code

```
sc stop <ServiceName>
```

3. Pause a Service

You can **pause** a service temporarily:

- **Right-click** on the service and select **Pause**.

Command-Line Equivalent:

bash

Copy code

```
sc control <ServiceName> pause
```

4. Resume a Paused Service

You can **resume** a paused service:

- **Right-click** on the service and select **Resume**.

Command-Line Equivalent:

bash

Copy code

```
sc control <ServiceName> continue
```

5. Restart a Service

You can **restart** a service (stop and start):

- **Right-click** on the service and select **Restart**.

Command-Line Equivalent:

bash

Copy code

```
net stop <ServiceName> && net start <ServiceName>
```

Or using **sc**:

bash

Copy code

```
sc stop <ServiceName> && sc start <ServiceName>
```

Service Configuration

6. Change Startup Type

You can change the **Startup Type** of a service to one of the following:

- **Automatic**: Starts the service when the system boots.
- **Manual**: Starts the service when needed by another process.
- **Disabled**: The service is completely disabled and will not start.

To change this:

- **Right-click** on the service, select **Properties**, and change the **Startup type** under the **General** tab.

Command-Line Equivalent: To change the startup type of a service:

bash

Copy code

```
sc config <ServiceName> start= <StartupType>
```

For example:

bash

Copy code

```
sc config <ServiceName> start= automatic
```

Where `<StartupType>` can be:

- `auto` (Automatic)
- `manual`
- `disabled`

---

7. View Service Dependencies

You can view the dependencies of a service (services that depend on this service, and services that this service depends on):

- **Right-click** on the service and select **Properties**, then go to the **Dependencies** tab.