# Server Operating Systems

## Lecture 6

Finding, Searching
and Sorting Files

## File Systems and File Utilities

- Partitions
- Mounting File Systems
- Disk Usage Commands
- Finding Files Using the Command Line - find
- Searching for Text Strings in Files - grep
- Sorting Files and Command Output - sort
- Editing Output - sed

# Partitions

A *partition* is a contiguous section of a disk that holds data.

Windows allows up to 4 primary partitions - to get past this limitation, they also allow an extended partion, which can contain approx 24 logical drives.

Solaris UNIX can have up to 8 partitions, or *slices*.

Linux supports a single partition with multiple logical partitions.

OS's refer to each partition or slice as an independent drive.

- Each one is associated with a drive name called a *mount point*.
- Located in /dev.

# Linux Partition Names

| Partition | Mount Point |
|---|---|
| /dev/hda1 | / |
| /dev/hda2 | /home |
| /dev/hda3 | swap |
| /dev/hdb1 | /usr/backup |
| /dev/hdb2 | /opt |

This shows two physical hard drives divided up into partitions.

sda, sdb is used for SATA, SCSI and USB disks

## File Systems

To the user, the file system is the hierarchy of files on disk.

To the OS, the file system is a structure on a partition that tells the OS where files are located, physically, on disk.

A partition does not need a file system on it, but it must have one to be usable by an end user.

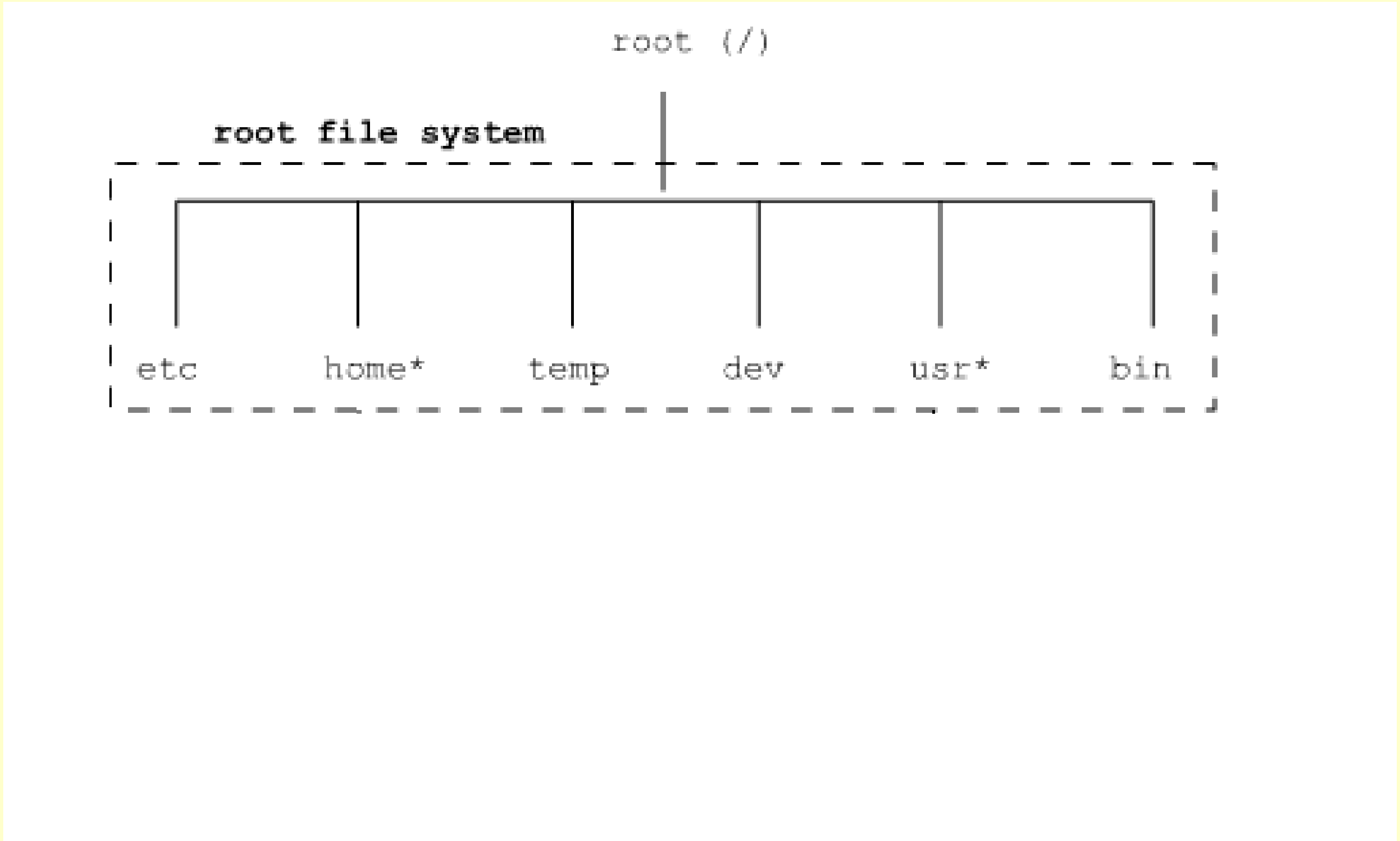
## Mounting the File System

All separate file systems (partitions) are combined at boot time to form a single file system.

Each partition is “mounted” to an empty directory, the *mount point*.

Directories and files in an unmounted file system are inaccessible.

## Mounting the File System



* Indicates an empty directory used as a mount point

## mount and umount

Most modern linux systems will automatically mount the filesystems on a CD or USB stick when it is inserted.

If not, there are two commands which allow the system administrator to do so:

```
mount -t ISO9660 /dev/cdrom /mnt/cd
mount -t vfat /dev/hda4 /media/windows

umount /mnt/cd
umount /dev/hda4
```

## File System Statistics

Use the `df` command to display system statistics.

- On Linux, size is listed in units of 1K.
- On Solaris UNIX, size is listed in units of .5K by default. Use `df –k` for 1K units.

The `du` command shows the disk space used by files and subdirectories.

- Use `du –k` to list sizes in units of 1K.
- Can use on a specific part of the file system: `du –k` ***`pathname`***.

## The `find` Command

Find files based on a set of criteria and optionally execute commands on the found files.

This command searches all files and subdirectories in the directory you are currently looking at.

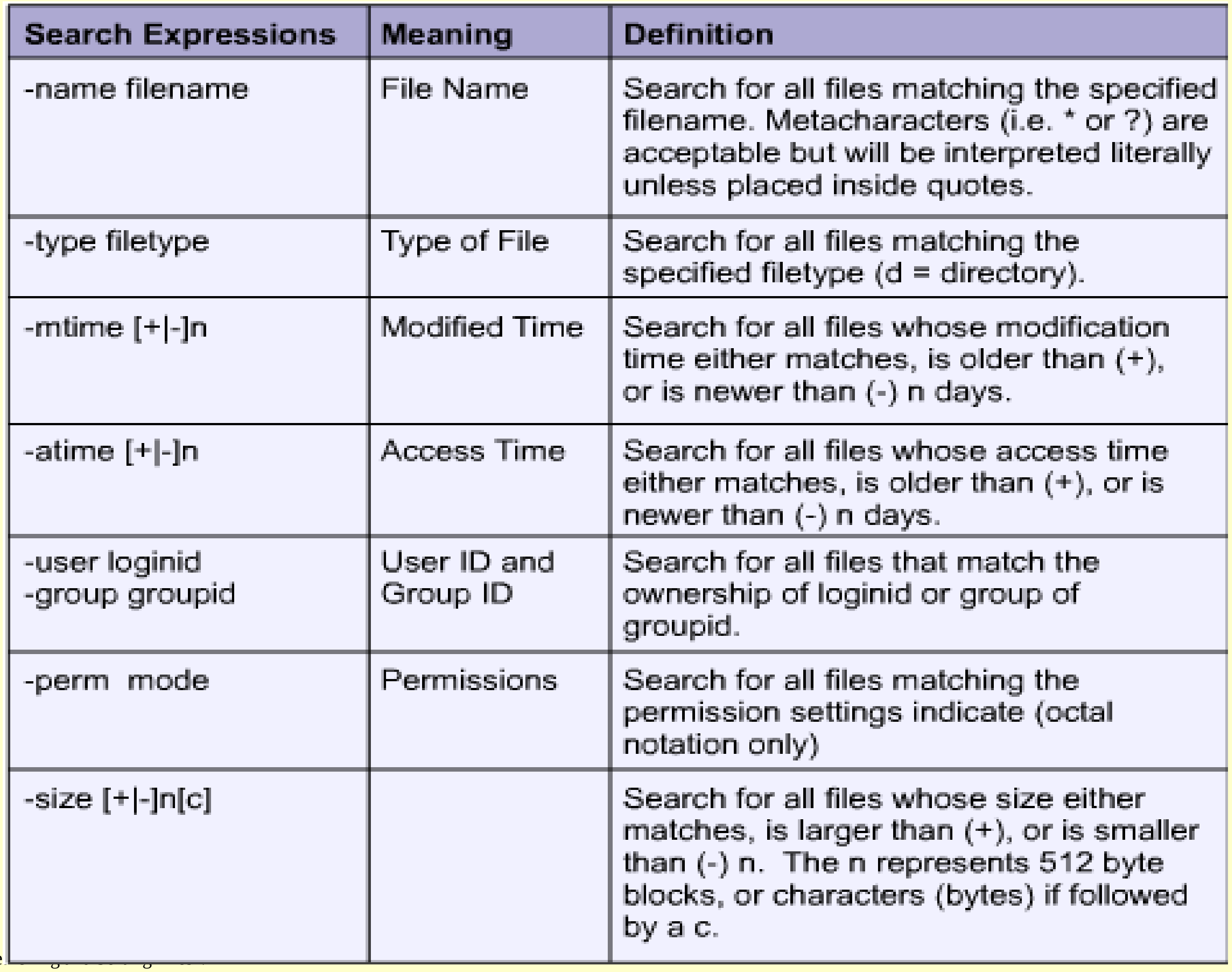
This can take a very long time, so you should start at the “lowest” point where you think a file may be located

## The `find` Command

**`find` *`path expression [action]`***

- `path` is where to start the search.
- `expression` is one or more search criteria that describes what to look for.
  - Evaluates to “true” or “false.”
  - Multiple expressions are considered to be a single *AND* expression.
  - `-o` can be used between expressions to *OR* them together.
- `action` is where commands can be specified to act on the found files.

# The `find` Command



| Search Expressions | Meaning | Definition |
|---|---|---|
| -name filename | File Name | Search for all files matching the specified filename. Metacharacters (i.e. * or ?) are acceptable but will be interpreted literally unless placed inside quotes. |
| -type filetype | Type of File | Search for all files matching the specified filetype (d = directory). |
| -mtime [+\|-]n | Modified Time | Search for all files whose modification time either matches, is older than (+), or is newer than (-) n days. |
| -atime [+\|-]n | Access Time | Search for all files whose access time either matches, is older than (+), or is newer than (-) n days. |
| -user loginid<br>-group groupid | User ID and Group ID | Search for all files that match the ownership of loginid or group of groupid. |
| -perm  mode | Permissions | Search for all files matching the permission settings indicate (octal notation only) |
| -size [+\|-]n[c] | | Search for all files whose size either matches, is larger than (+), or is smaller than (-) n.  The n represents 512 byte blocks, or characters (bytes) if followed by a c. |

## Finding Files

```
find /usr -name openwin
```

Search for files called openwin in directory /usr

```
find /home/user01 -name '*tif'
```

Search for files whose names end in 'tif' in directory /home/user01

```
find dir3 -type d
```

Search for files of type 'directory' starting in directory dir3

```
find . -mtime +90
```

Search for files starting at current directory that have not been modified in the last 90 days

```
find ~ -size +400
```

Search for files larger than 400 blocks starting in your home directory.

```
find ~ -perm 777 > holes.txt
```

Search for files that have open permissions in your home directory, and send the list of names to holes.txt

## The `grep` Command

"Global Regular Expression Print"

Searches for text *inside* of a file or within the output of another command.

Works with strings (one or more words) or *regular expressions*.

Regular Expressions are a complex set of rules which define wildcard characters which can be used to search for text.

You are not expected to understand the details of Regular Expressions for this course.

## The `grep` Command

**`grep [`*`option(s)`*`]`** ***`string filename`***

**`string`** can be a character, a word, or a sentence.

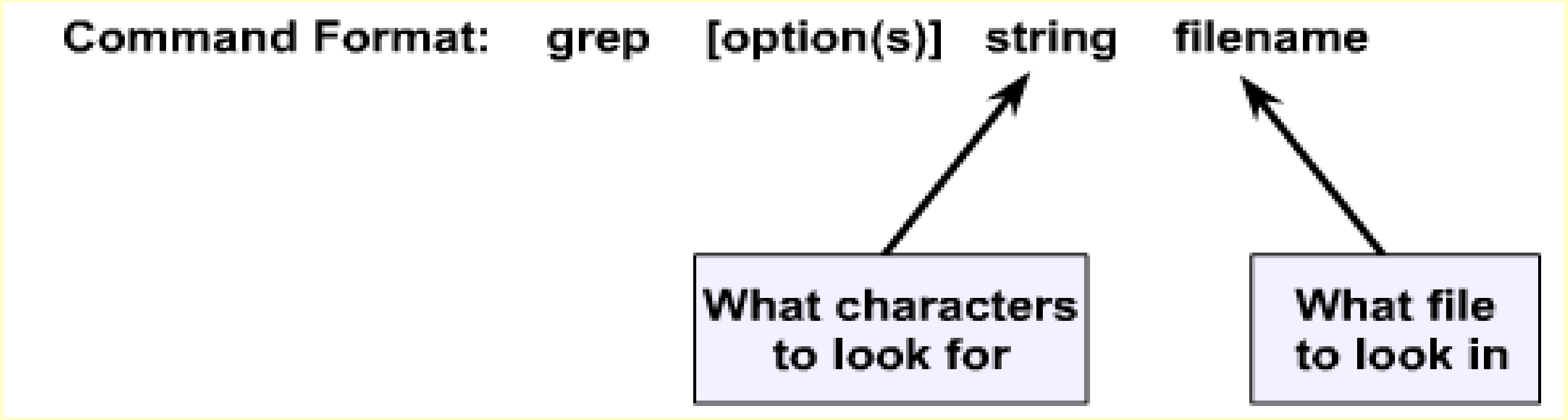
A **`string`** with whitespace or punctuation in it must be surrounded by quotation marks '   '.

**`-i`** ignores case (case-sensitive by default).

**`-v`** prints out only lines that *do not* match **`string`.**

**`-n`** displays line numbers.

# grep Command


Command Format: grep [option(s)] string filename
What characters to look for
What file to look in

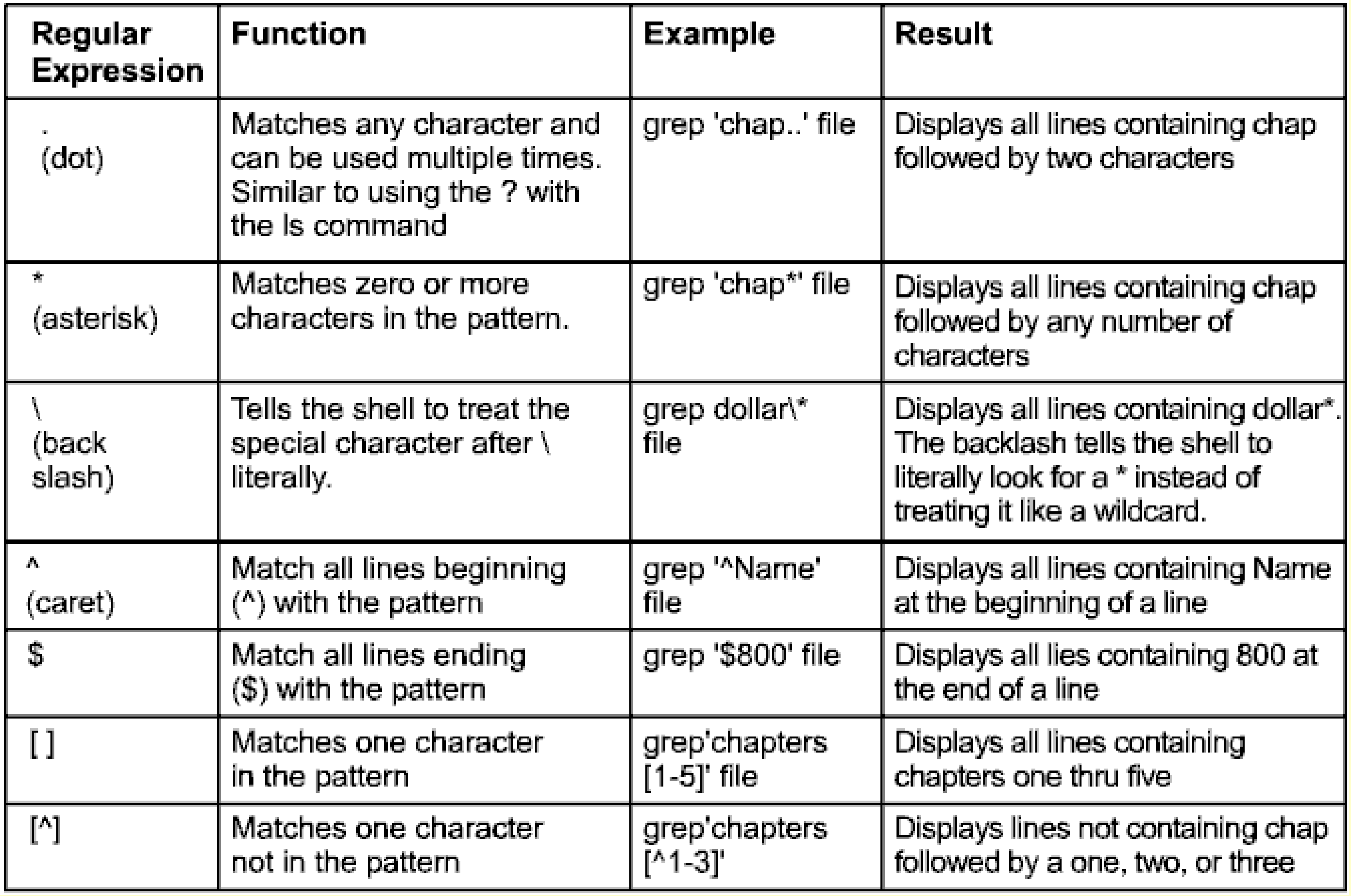
search for pattern "rose” inside of text file “flowers”

```
grep rose flowers
```

looks for who is on server by piping who output to pattern search for user2

```
who | grep  user2
```

# Common Regular Expressions



| Regular Expression | Function | Example | Result |
|---|---|---|---|
| . (dot) | Matches any character and can be used multiple times. Similar to using the ? with the ls command | grep 'chap..' file | Displays all lines containing chap followed by two characters |
| * (asterisk) | Matches zero or more characters in the pattern. | grep 'chap*' file | Displays all lines containing chap followed by any number of characters |
| \ (back slash) | Tells the shell to treat the special character after \ literally. | grep dollar* file | Displays all lines containing dollar*. The backlash tells the shell to literally look for a * instead of treating it like a wildcard. |
| ^ (caret) | Match all lines beginning (^) with the pattern | grep '^Name' file | Displays all lines containing Name at the beginning of a line |
| $ | Match all lines ending ($) with the pattern | grep '$800' file | Displays all lies containing 800 at the end of a line |
| [ ] | Matches one character in the pattern | grep'chapters [1-5]' file | Displays all lines containing chapters one thru five |
| [^] | Matches one character not in the pattern | grep'chapters [^1-3]' | Displays lines not containing chap followed by a one, two, or three |

# The `sort` Command

Sort the contents of a file based on *fields*.

- By default, fields are delimited by whitespace.
- Sorting moves from left-to-right, character by character.
- ASCII ordering scheme.

```
sort [option(s)] [input_filename]
```

**+3**: Begin sort on the field *following* the 3rd field.

**-2**: End sort on the field *following* the 2nd field.

**-k 4**: Sort on 4th field.

```
001  Kevin Wilson
002  Paul  Jones
003  Helen Thorpe
004  Pat   Hunter

sort +2 filetest          will give

004  Pat   Hunter
002  Paul  Jones
003  Helen Thorpe
001  Kevin Wilson
```

## File Editing with Sed

The “Stream Editor.”

Sed reads lines from a file sequentially and applies user specified editing commands to each one.

Sed is “non-destructive,” it doesn’t change the contents of the original file, because it sends its output to stdout.

- You have to redirect Sed’s output to another file to save changes.

# File Editing with Sed

```
sed [option(s)] [command] filename [>newfile]
or
command | sed [option(s)] [command]
```

Use the `–n` option to suppress printing lines out.

sed has it's own command language - it can also use the regular expression rules that are used with grep.

| | |
|---|---|
| a\ | append text |
| c\ | replace text |
| i\ | insert text |
| d | delete lines |
| s | search and substitute |
| /regexp/ | apply the regexp that is contained in slashes |

# sed Examples

```
sed –n '20,25p' file
```

Print only lines 20 to 25 of file

```
sed '5d' file
```

Delete line 5 from file

```
ls –l | sed '5,$d' > newfile
```

Take the directory listing, delete lines 5 to the last line and put the result in newfile

```
sed '1,10s/Windows/UNIX/g' file
```

Search the first 10 lines of file, and globally replace 'Windows' with 'UNIX'.