```
** BASIC GREP **

# Search for a pattern in a file
grep 'pattern' filename

# Case-insensitive search
grep -i 'pattern' filename

# Search recursively in a directory
grep -r 'pattern' /path/to/dir

# Show line numbers with matches
grep -n 'pattern' filename

# Invert match (show lines that do NOT match)
grep -v 'pattern' filename

# Show only matched strings, not the whole line
grep -o 'pattern' filename

# Search for whole words only
grep -w 'word' filename

# Count number of matching lines
grep -c 'pattern' filename

# Match multiple patterns (OR)
grep -E 'pattern1|pattern2' filename

# Match lines starting with a pattern
grep '^pattern' filename

# Match lines ending with a pattern
grep 'pattern$' filename

# Display lines before/after/around match
grep -B 3 'pattern' filename  # 3 lines before
grep -A 3 'pattern' filename  # 3 lines after
grep -C 2 'pattern' filename  # 2 lines before & after



** ADVANCED GREP / REGEX **

# Match digits
grep '[0-9]' filename

# Match letters
grep '[a-zA-Z]' filename

# Match a range of letters or numbers
grep '[a-d0-5]' filename

# Match zero or more of previous character
grep 'ab*' filename  # a followed by zero or more b's

# Match one or more of previous character
grep 'ab\+' filename

# Match optional character
grep 'ab\?' filename

# Match exactly N occurrences
grep -E 'a{3}' filename  # three consecutive a's
```

```
# Match N to M occurrences
grep -E 'a{2,5}' filename

# Match start/end of line
grep '^start' filename
grep 'end$' filename

# Match any character
grep 'a.c' filename  # matches 'abc', 'a1c', etc.

# Word boundaries
grep -E '\<word\>' filename



** SED BASIC **

# Print a file (default output)
sed '' filename

# Substitute first occurrence on a line
sed 's/old/new/' filename

# Substitute all occurrences on a line
sed 's/old/new/g' filename

# Case-insensitive substitution
sed 's/old/new/Ig' filename

# Substitute only on lines matching a pattern
sed '/pattern/s/old/new/g' filename

# Delete lines matching a pattern
sed '/pattern/d' filename

# Delete lines NOT matching a pattern
sed '/pattern/!d' filename

# Print only lines matching a pattern
sed -n '/pattern/p' filename

# Print line numbers with matches
sed -n '/pattern/=' filename

# Insert a line before a match
sed '/pattern/i\New line text' filename

# Append a line after a match
sed '/pattern/a\New line text' filename

# Replace text using multiple commands
sed -e 's/old1/new1/g' -e 's/old2/new2/g' filename



** SED ADVANCED **

# Replace in-place (overwrite file)
sed -i 's/old/new/g' filename

# Replace in-place with backup
sed -i.bak 's/old/new/g' filename
```

```
# Only replace on specific line numbers
sed '3s/old/new/' filename  # line 3
sed '1,5s/old/new/g' filename  # lines 1-5

# Delete specific lines by number
sed '5d' filename   # delete line 5
sed '3,7d' filename # delete lines 3-7

# Print specific line numbers
sed -n '5p' filename   # print line 5
sed -n '3,7p' filename # print lines 3-7

# Read commands from a file
sed -f commands.sed filename

# Use & to represent matched text
sed 's/old/& and new/' filename  # 'old' becomes 'old and new'

# Use grouping and backreferences
sed -r 's/(foo)(bar)/\2\1/' filename  # swaps 'foobar' → 'barfoo'

# Multiple-line pattern space (N command)
sed 'N;s/foo\nbar/foobar/' filename  # combine two lines

# Delete empty lines
sed '/^$/d' filename

# Trim leading/trailing whitespace
sed 's/^[ \t]*//;s/[ \t]*$//' filename

# Transform characters (upper/lower)
sed 'y/abcdefghijklmnopqrstuvwxyz/ABCDEFGHIJKLMNOPQRSTUVWXYZ/' filename

# Conditional commands with pattern
sed '/pattern/{s/old/new/g;d}' filename  # if match, substitute and delete line

** SED + GREP COMBINATIONS **

# Search and replace only matching lines
grep 'pattern' file | sed 's/foo/bar/g'

# Count occurrences of a pattern
grep -o 'pattern' file | wc -l

# Highlight matches in terminal
grep --color=always 'pattern' file | less -R



** USEFUL FLAGS / SHORTCUTS **

# Show help for grep
grep --help

# Show help for sed
sed --help

# Debug sed scripts
sed -n -e 's/old/new/p' filename  # prints substitutions only



** REGEX BASICS **
```

```
# Dot matches any character except newline
.

# Start of line
^

# End of line
$

# Zero or more occurrences of previous character
*

# One or more occurrences of previous character
+

# Optional (0 or 1) occurrence
?

# Exact number of occurrences
{n}

# Range of occurrences
{n,m}

# Character classes (digits, letters)
[0-9], [a-zA-Z], [a-d0-5]

# Negated character class
[^0-9]

# Word boundary
\b

# Non-word boundary
\B

# Alternation (OR)
a|b

# Grouping
(foo|bar)

# Escape special character
\., \*, \$, \^

# Match whitespace / non-whitespace
\s, \S

# Match digit / non-digit
\d, \D

# Match word / non-word
\w, \W

# Match tab / newline
\t, \n

# Lookahead / Lookbehind (if supported)
(?=pattern), (?<=pattern)
```