LINUX COMMANDS

```
date # Displays the current date and time.

date "+%D" # Shows the date in MM/DD/YY format.

date "+%D %T" # Shows date and time.

date "+%Y-%m-%d" # Displays date in YYYY-MM-DD format.

date --date "tomorrow" # Shows the date for tomorrow.

date --date "2 years ago" # Displays the date from two years ago.

man date # Opens manual for date.

bc # Opens command-line calculator.

bc <<< "12/5" # Performs division using bc.

bc <<< "scale=2; 12/5" # Division with 2 decimal precision.

bc <<< "2>5" # Checks if 2 is greater than 5.

bc <<< "2^10" # Calculates 2 raised to power 10.

uname # Shows system information.

uname -a # Displays all system info.

uname -r # Shows kernel version.

uname -s # Shows kernel name.

uname -n # Displays network hostname.

sudo apt update # Updates the package list.

sudo apt upgrade # Upgrades all upgradable packages.

sudo apt install gcc # Installs GCC compiler.

sudo apt install build-essential # Installs essential development tools.

sudo apt install ncal # Installs the ncal calendar utility.

ls # Lists files and directories.

ls -l # Lists with details.

ls -a # Lists including hidden files.

ls -R # Recursively lists directories.

cd dir # Changes to directory dir.

cd .. # Goes one level up.

cd - # Switches to previous directory.

mkdir dir # Creates a directory.

mkdir -p d1/d2 # Creates nested directories.
```

```
pwd # Prints current working directory.

echo "text" # Prints text.

echo -e "Hello \t World" # Prints text with tab space.

echo -n "Hello" # Prints without new line.

printf "text\n" # Prints formatted text.

cat file # Displays file content.

cat -n file # Shows file with line numbers.

tac file # Displays file content in reverse.

rev <<< text # Reverses the text.

nano file # Opens file in Nano editor.

vi file # Opens file in Vi editor.

touch file # Creates a new empty file.

cp file dest # Copies file to destination.

cp -r dir dest # Recursively copies directory.

mv file dest # Moves/renames file.

rm file # Deletes file.

rm -r dir # Deletes directory and contents.

rmdir dir # Removes empty directory.

ps # Displays current running processes.

exit # Exits the terminal session.

echo $SHELL # Shows default shell.

printenv SHELL # Prints shell variable.

whoami # Displays current username.

whatis command # Shows a short description of command.

whereis command # Shows location of binary, source, and man pages.

info command # Displays command's info manual.

man command # Opens command manual page.

date && whoami # Runs date, then whoami if successful.

echo $? # Shows exit status of the last command.

false # Returns a false (non-zero) exit status.

cal # Displays calendar.

seq 1 10 # Prints numbers from 1 to 10.

seq 1 3 10 # Prints numbers from 1 to 10 in steps of 3.
```

```
history # Shows command history.
gcc file.c && ./a.out # Compiles and runs a C program.
vi sum.c                      # Opens 'sum.c' file in the vi editor.
gcc sum.c && ./a.out           # Compiles 'sum.c' and runs the output if compilation is successful.
vi filecmd                   # Opens 'filecmd' file in the vi editor.
cat filecmd                  # Displays the contents of 'filecmd'.
cat sum.c                    # Displays the contents of 'sum.c'.
gcc sum.c                    # Compiles the C program in 'sum.c'.
./a.out                      # Executes the compiled output file 'a.out'.
gcc sum.c && ./a.out         # Same as line 324.
ls                           # Lists files and directories in the current folder.
more filecmd                 # Views 'filecmd' content page by page.
nano filecmd                 # Opens 'filecmd' in the nano editor.
more filecmd                 # Views 'filecmd' content again.
less filecmd                 # Views 'filecmd' with navigation support (scrolling).
head filecmd                 # Displays the first 10 lines of 'filecmd'.
head -n 5 filecmd            # Displays the first 5 lines of 'filecmd'.
tail filecmd                 # Displays the last 10 lines of 'filecmd'.
tail +5 filecmd              # Displays content from line 5 onward in 'filecmd'.
nano sample.txt              # Opens or creates 'sample.txt' in nano.
cat sample.txt               # Displays contents of 'sample.txt'.
cut -c 1-3 sample.txt        # Extracts characters 1 to 3 from each line.
cut -c 4-10 sample.txt       # Extracts characters 4 to 10 from each line.
cat sample.txt               # Displays contents again.
cut -d ' #' -f2 sample.txt    # Cuts and shows the 2nd field using ':' as delimiter.
cut -d ' #' -f2,3 sample.txt  # Cuts and shows 2nd and 3rd fields.
ls                           # Lists files again.
cat demo                     # Displays contents of 'demo'.
nano f1                      # Creates/edits file 'f1' in nano.
nano f2                      # Creates/edits file 'f2' in nano.
cat fi                       # Likely typo; tries to show contents of 'fi'.
cat f1                       # Displays 'f1'.
cat f2                       # Displays 'f2'.
paste f1 f2                  # Merges 'f1' and 'f2' line by line.
paste -d ' #' f1 f2           # Same as above but uses ':' as delimiter.
paste -s f1 f2               # Pastes content serially (one file after another).
cat sample.txt               # Shows 'sample.txt'.
sort sample.txt              # Sorts lines alphabetically.
sort sample.txt -r           # Sorts lines in reverse order.
sort -t ' #' -k2 sample.txt   # Sorts by 2nd field using ':' delimiter.
sort -t ' #' -k3 sample.txt   # Sorts by 3rd field.
sort -t ' #' -k4 sample.txt   # Sorts by 4th field.
sort -t ' #' -k3 sample.txt -r # Reverse sort by 3rd field.
cat sample.txt               # Displays again.
tr ' #' '|' < sample.txt      # Replaces ':' with '|' in output.
cat sample.txt               # Displays unchanged file.
ls                           # Lists files again.
tr ' #' '|' < sample.txt > s1.txt # Saves output of ':' replaced with '|' to 's1.txt'.
ls                           # Lists to confirm 's1.txt'.
cat s1.txt                   # Shows contents of 's1.txt'.
tr ' #0' '|$' < sample.txt    # Replaces ':' with '|' and '0' with '$' (though incorrect syntax).
cat sample.txt               # Displays again.
tr -s '0' < sample.txt       # Squeezes repeated 0s into a single 0.
cat sample.txt               # Displays again.
tr -d '0' < sample.txt       # Deletes all 0s from output.
nano example.txt             # Edits/creates 'example.txt'.
cat example.txt              # Displays contents.
uniq example.txt             # Removes adjacent duplicate lines.
nano example.txt             # Edits again.
cat example.txt              # Displays again.
uniq example.txt             # Shows unique lines again.
ls                           # Lists all files.
cat sample.txt               # Displays again.
cat s1.txt                   # Displays again.
```

```
cmp sample.txt s1.txt        # Compares files byte-by-byte.
diff sample.txt s1.txt       # Shows line-by-line differences.
ls                           # Lists files.
nano test.c                  # Opens 'test.c' for editing.
cat test.c                   # Shows code in 'test.c'.
grep main test.c             # Finds lines with 'main' in test.c.
grep ^main test.c            # Finds lines that start with 'main'.
grep ^int test.c             # Finds lines starting with 'int'.
grep \; test.c               # Finds lines containing a semicolon.
nano test.c                  # Edits again.
cat test.c                   # Displays again.
grep ^int test.c             # Finds 'int' starting lines again.
nano sed_test                # Opens/creates 'sed_test'.
cat sed_test                 # Displays file.
sed 's/Hello/Hi/' sed_test   # Replaces first 'Hello' with 'Hi' per line (not in file).
cat sed_test                 # File unchanged.
sed -i 's/Hello/Hi/' sed_test # Replaces in-place (modifies file).
cat sed_test                 # Shows updated content.
sed 's/!/$/g' sed_test       # Replaces all '!' with '$'.
cat sed_test                 # Shows file.
sed '/simple/d' sed_test     # Deletes lines containing 'simple'.
sed '/Hi/a Welcome to sed' sed_test # Appends line after 'Hi'.
sed -n '/Hi/p' sed_test      # Prints only lines containing 'Hi'.
nano employees.txt           # Creates/edits 'employees.txt'.
cat employees.txt            # Displays employee data.
awk '{print $0}' employees.txt # Prints every line (entire record).
awk '{print $1, $3}' employees.txt # Prints 1st and 3rd columns.
awk $3>50000 '{print $1, $3}' employees.txt # Invalid syntax (error).
awk '$3 > 50000 {print $1, $3}' employees.txt # Prints names and salary > 50000.
awk 'BEGIN {print "Name #, Salary: "} {print $1, $3}' employees.txt # Prints headers and names/salaries.
awk 'BEGIN {print "Name Salary"} {print $1, $3}' employees.txt # Prints with proper header.
awk 'BEGIN {print "Name Age Salary"} {print $1, $2, $3}' employees.txt # Prints all columns with header.
awk '{total+= $3} END {print "Total Salary=", total}' employees.txt # Sums and prints total salary.
awk '{print $0}' employees.txt # Prints entire file (same as line 410).
awk '{print NR, $1}' employees.txt # Prints line number and name.
awk '{print NR, $0}' employees.txt # Prints line number with full line.
awk '/Alice/' employees.txt # Prints lines containing 'Alice'.
awk '$2 < 30 {print $1, $2}' employees.txt # Prints name and age if age < 30.
awk '{printf "Name # %s, Age: %d, Salary: %d\n", $1, $2, $3}' employees.txt # Formatted output.
awk 'END {print "Total employees= ", NR}' employees.txt # Prints total line count (employees).
ls -l                        # Long list with permissions and details.
ls -l s1.txt                 # Shows details of 's1.txt'.
chmod u+x s1.txt             # Gives execute permission to user for 's1.txt'.
ls -l s1.txt                 # Verifies permission change.
chmod o-r s1.txt             # Removes read permission from others.
ls -l s1.txt                 # Verifies again.
chmod 462 s1.txt             # Sets specific file permissions.
ls -l s1.txt                 # Verifies final permission state.
history                      # Shows command history.
```