

BASH QUICK REFERENCE

Commands, scripting, pipes, redirection, job control

Basics

echo & Navigation

```
echo "Hello, World!" # print text
pwd                 # print working directory
cd /path/to/dir    # change directory
cd ..              # go up one level
cd -               # go to home directory
cd -               # go to previous directory
```

Listing & Creating

```
ls                 # list files
ls -la            # long format, show hidden
ls -lh           # human-readable sizes
mkdir mydir      # create directory
mkdir -p a/b/c   # create nested directories
```

Copy, Move & Remove

```
cp file.txt copy.txt # copy file
cp -r dir/ backup/   # copy directory recursively
mv old.txt new.txt   # rename / move
rm file.txt          # delete file
rm -r dir/           # delete directory recursively
rm -rf dir/          # force delete (no prompt)
```

Variables & Expansion

Variables

```
name="Alice"       # assign (no spaces!)
echo "$name"       # variable expansion
echo "${name}_file" # braces for clarity
readonly PI=3.14    # constant
unset name          # delete variable
```

Special Variables

```
$_ Script name
$1 $2 ... Positional arguments
 $# Number of arguments
@ All arguments (separate words)
* All arguments (single string)
$? Exit status of last command
$$ Current process ID
$_ PID of last background process
```

Command Substitution & Arithmetic

```
files=$(ls) # capture output
today=$(date +%Y-%m-%d) # command substitution
count=$((5 + 3)) # arithmetic: 8
echo $(10 / 3) # integer division: 3
echo ${10 % 3} # modulo: 1
```

String Operations

```
 ${#str} String length
 ${str:0:5} Substring (offset:length)
 ${str/old/new} Replace first match
 ${str//old/new} Replace all matches
 ${str^^} Uppercase
 ${str,,} Lowercase
```

Conditionals

if / elif / else

```
if [[ "$name" == "Alice" ]]; then
  echo "Hi Alice"
elif [[ "$name" == "Bob" ]]; then
  echo "Hi Bob"
else
  echo "Who are you?"
fi
```

Test Operators

```
-eq -ne Integer equal / not equal
-lt -gt Integer less / greater than
-le -ge Integer less/greater or equal
== != String equal / not equal
-z "$str" String is empty
-n "$str" String is not empty
-f file File exists and is regular
-d dir Directory exists
-e path Path exists (any type)
-r -w -x Readable / writable / executable
&& || Logical AND / OR
```

Loops

for Loop

```
for fruit in apple banana cherry; do
  echo "$fruit"
done

for f in *.txt; do
  echo "File: $f"
done
```

C-style for Loop

```
for ((i=0; i<5; i++)); do
  echo "$i"
done
```

while Loop

```
count=0
while [[ $count -lt 5 ]]; do
  echo "$count"
  ((count++))
done
```

Loop Control

```
break Exit the loop
continue Skip to next iteration
```

Functions

Defining & Calling

```
greet() {
  echo "Hello, $1!" # $1 = first arg
  return 0          # exit status
}
greet "Alice"      # Hello, Alice!
```

Local Variables & Return Values

```
add() {
  local sum=$(( $1 + $2 ))
  echo "$sum" # "return" via stdout
}
result=$(add 3 5) # capture: 8
```

Pipes & Redirection

Pipes

```
ls -l | grep ".txt" # pipe output
cat log | sort | uniq # chain commands
cmd1 | tee out.txt # pipe + save to file
```

Redirection

```
cmd > file Redirect stdout (overwrite)
cmd >> file Redirect stdout (append)
cmd < file Redirect stdin from file
cmd 2> file Redirect stderr
cmd 2>&1 Redirect stderr to stdout
cmd &> file Redirect stdout + stderr
cmd << EOF Here document (inline input)
/dev/null Discard output: `cmd >/dev/null`
```

File Operations

Viewing Files

```
cat file.txt # print entire file
head -n 10 file.txt # first 10 lines
tail -n 10 file.txt # last 10 lines
tail -f log.txt # follow (live updates)
less file.txt # paginated viewer
```

Counting & Finding

```
wc -l file.txt # count lines
wc -w file.txt # count words
wc -c file.txt # count bytes
find . -name "*.txt" # find by name
find . -type d # find directories
find . -mtime -7 # modified in last 7 days
```

Other File Commands

```
touch file Create file / update timestamp
stat file File metadata (size, dates)
file img.png Detect file type
diff a.txt b.txt Compare two files
sort file.txt Sort lines
uniq Remove adjacent duplicates
cut -d: -f1 Extract fields by delimiter
tr 'a-z' 'A-Z' Translate / replace characters
```

Text Processing

grep

```
grep "error" log.txt # search for pattern
grep -i "error" log.txt # case-insensitive
grep -r "TODO" src/ # recursive search
grep -n "func" file.go # show line numbers
grep -c "error" log.txt # count matches
grep -v "debug" log.txt # invert match
```

sed

```
sed 's/old/new/' file # replace first per line
sed 's/old/new/g' file # replace all
sed -i 's/old/new/g' file # edit in place
sed -n '5,10p' file # print lines 5-10
sed '/pattern/d' file # delete matching lines
```

awk

```
awk '{print $1}' file # print first field
awk -F: '{print $1}' file # custom delimiter
awk 's3 > 100' file # filter by field value
awk '{sum+=1} END{print sum}' file # sum column
```

Permissions

chmod

```
chmod 755 script.sh # rwxr-xr-x
chmod +x script.sh # add execute
chmod -w file.txt # remove write
chmod u+x,g-w file # user +exec, group -write
```

Permission Reference

```
r (4) Read
w (2) Write
x (1) Execute
u / g / o User / Group / Others
755 Owner: rwx, Group/Other: r-x
644 Owner: rw-, Group/Other: r--
```

Ownership

```
chown user:file.txt # change owner
chown user:group file.txt # change owner + group
chown -R user:group dir/ # recursive
```