Bash Cheat Sheet: Easy Memory Edition (with Examples)

1. Files: "Is it there? What is it?"

- **-d**: **D**irectory?
  - if [-d "/home/user/Documents"]; then echo "Documents is a directory"; fi
- **-f**: **F**ile (regular)?
  - if [-f "myfile.txt"]; then echo "myfile.txt is a file"; fi
- **-e**: **E**xists?
  - if [-e "config.ini"]; then echo "config.ini exists"; fi
- **-rwx**: **R**ead, **W**rite, e**X**ecute? (Specifically -r, -w, -x)
  - if [-r "script.sh"]; then echo "script.sh is readable"; fi
  - if [-w "data.log"]; then echo "data.log is writable"; fi
  - if [-x "program"]; then echo "program is executable"; fi
- **-s**: **S**ize (non-empty)?
  - if [-s "report.txt"]; then echo "report.txt is not empty"; fi

2. Strings: "Compare and Check"

- **-z**: **Z**ero (empty)?
  - if [-z "$MYVAR"]; then echo "MYVAR is empty"; fi
- **-n**: **N**ot zero (not empty)?
  - if [-n "$MYVAR"]; then echo "MYVAR is not empty"; fi
- **==**: Equal?
  - if ["$USER" == "root"]; then echo "You are root"; fi
- **!=**: Not equal?
  - if ["$OS" != "Windows"]; then echo "Not a Windows system"; fi
- **< >**: Less/Greater (alphabetical)?
  - if [["apple" < "banana"]]; then echo "apple comes before banana"; fi

3. Numbers: "Math Comparisons"

- **-eq**: **E**qual?
  - if ["$COUNT" -eq 10]; then echo "COUNT is 10"; fi
- **-ne**: **N**ot equal?
  - if ["$AGE" -ne 0]; then echo "AGE is not zero"; fi
- **-lt, -le**: **L**ess than, **L**ess than or equal?
  - if ["$NUM" -lt 5]; then echo "NUM is less than 5"; fi
  - if ["$SCORE" -le 100]; then echo "SCORE is less than or equal to 100"; fi
- **-gt, -ge**: **G**reater than, **G**reater than or equal?
  - if ["$SIZE" -gt 1024]; then echo "SIZE is greater than 1024"; fi
  - if ["$VERSION" -ge 2]; then echo "VERSION is 2 or higher"; fi

4. Logic: "Combine and Negate"

- **&&**: AND (both must be true)
  - if [-d "mydir"] && [-w "mydir"]; then echo "mydir exists and is writable"; fi
- **||**: OR (one or both true)
  - if [-f "file1.txt"] || [-f "file2.txt"]; then echo "Either file1.txt or file2.txt exists"; fi
- **!**: NOT (reverse truth)
  - if [! -f "temp.txt"]; then echo "temp.txt does not exist"; fi

5. Special Variables: "Info About the Script"

- **$?**: Last command's **?** (exit code)
  - ls non_existent_file; echo "Exit code: $?"
- **$$**: My **P**rocess **I**dentifier (PID)
  - echo "Script PID: $$"

- **$#**: **#** of arguments
    - echo "Number of arguments: $#"
- **$@**: All arguments (separate)
    - for arg in "$@"; do echo "Argument: $arg"; done

6. Redirects: "Send Input/Output"

- **>**: Overwrite output
    - ls > filelist.txt
- **>>**: Append output
    - echo "New data" >> data.log
- **<**: Input from file
    - wc -l < input.txt
- **2>**: Error output
    - command_that_might_fail 2> errors.log
- **&>**: All output (errors and normal)
    - script.sh &> output.log

7. Process Substitution: "Command Output as File"

- **<()**: Use command output as input
    - diff <(ls dir1) <(ls dir2)
- **>()**: Use file descriptor as output
    - tee >(gzip > output.gz) < input.txt

8. Brace Expansion: "Generate Lists"

- **{1..5}**: Numbers 1 to 5
    - echo {1..5}
- **{a..e}**: Letters a to e
    - echo {a..e}
    - touch file{1..3}.txt

9. Misc. "Extras"

- **&**: Run in background
    - long_running_command &
- ;: Multiple commands on one line
    - cd mydir; ls -l
- **&&**: If previous succeeds
    - mkdir newdir && cd newdir

10. Set Flags: "Shell Behavior"

- **-e**: Exit on error
    - set -e; command_that_might_fail; echo "This won't print if the above fails"
- **-x**: Show commands as they run
    - set -x; ls -l; echo "Done"; set +x
- **-u**: Error on unset variable
    - set -u; echo "$undefined_var"