Labwork 7 – Terminal Basics & Command-Line Fluency

Reference the provided `Linux_cheatsheet_shortx.pdf` for flags and examples; use any additional man pages you need (`man`, `--help`). Capture the commands you ran and short notes on what each did. Unless stated, work in a scratch directory and avoid destructive actions on real files.

Part A — Orientation & Navigation

1. Print your current working directory and the contents of the directory, including hidden files. Which flags showed the hidden entries?
2. Create the nested folders `projects/lab7/tmp` in one command. Verify they exist with `ls` using a format that shows trailing slashes for directories.
3. Jump one directory up, then back down into `projects/lab7/tmp` using relative paths only. Record the exact commands used.
4. From `projects`, list all `.md` files within the tree (max depth 2) without changing directories.

Part B — Creating & Inspecting Files

5. Create a file `notes.txt` containing two lines: `hello terminal` and `labwork seven`. Show two ways to do this (e.g., `cat >` vs `printf`).
6. Append a third line `appended line` to `notes.txt` without retyping the existing content. Confirm with `tail -n 2`.
7. Show the byte size and line count of `notes.txt` using a single command. Explain what each number in the output means.
8. View `notes.txt` with both `cat -n` and `nl`. Note any formatting differences.

Part C — Redirection & Pipelines

9. Using a single pipeline, sort the lines of `notes.txt` alphabetically, remove duplicates, and write the result to `sorted.txt` (don't overwrite `notes.txt`). Show the exact pipeline.
10. Create a command that counts how many unique words appear in `notes.txt` (case-insensitive). Keep this to one line with pipes.
11. Redirect only stderr from a failing command (e.g., `ls` on a nonexistent file) into `errors.log` while keeping stdout on screen. Show the resulting `errors.log` content.

Part D — Search & Filtering

12. Recursively search for the string `TODO` under the current directory, ignoring hidden folders. Record the command and a sample hit (path + line).
13. Find all files larger than 10 KB under `projects` and print their paths only. Avoid showing directories.
14. Using `grep`, display only the lines in `notes.txt` that contain the substring `line`, along with their line numbers.

Part E — Permissions & Execution

15. Display the long listing (`ls -l`) for `notes.txt` and explain the permission bits.
16. Change `notes.txt` to be readable and writable only by you (no permissions for group/others). Verify the change.
17. Create a short shell script `hello.sh` that echoes `Hello from Lab 7`, make it executable, and run it two ways: `./hello.sh` and `bash hello.sh`. Note the permission needed for the first method.

Part F — Archives & Compression

18. Create a tarball `lab7.tar.gz` containing `notes.txt`, `sorted.txt`, and `hello.sh`. List the contents of the archive without extracting.
19. Extract `lab7.tar.gz` into a new directory `lab7_extract` without overwriting existing files. Record the command used.

Part G — Processes & Jobs

20. Launch a `sleep 30` process in the background. Show how you list background jobs and bring it back to the foreground.
21. Start another `sleep 30` process, find its PID with `ps` (or an equivalent), and terminate it using `kill`. Confirm it ended.

Part H — Environment & Shell Shortcuts

22. Print your shell's `$PATH` split onto separate lines (one directory per line) using a pipeline.
23. Set an environment variable `LAB7_USER` for the current shell session and confirm it is visible to a child process (e.g., `env | grep LAB7`). How would you make it persistent across sessions?
24. Create an alias `ll` that expands to `ls -lah`. Demonstrate its use, then remove the alias.

Part I — Optional Stretch

25. Use `sed` or `awk` to prepend line numbers to `notes.txt` and write the result to `numbered.txt`.
26. Compress `notes.txt` with `gzip`, note the size difference, then decompress back to the original. Show the before/after `ls -l` output.
27. Use command substitution to store today's date in a variable and append a line `dated entry: <date>` to `notes.txt` in one command.

Deliverables

- A log of commands run and short answers (one to two sentences per question).
- The files you created (`notes.txt`, `sorted.txt`, `hello.sh`, `lab7.tar.gz`, and any optional artifacts) unless your instructor directs otherwise.
- Any questions or stumbling blocks you encountered while using the cheat sheet or man pages.