Section A: Conceptual Questions

1. What is the difference between Linux and Unix?

2. Explain the Linux file system hierarchy with examples.

3. What are absolute and relative paths in Linux?

4. What does each field in `-rw-r--r--` mean?

5. What is the significance of the first character in `ls -l` output?

6. Define the roles of owner, group, and others in Linux permissions.

7. What is the difference between `chmod 755` and `chmod 700`?

8. How is a symbolic link different from a hard link?

9. What is the purpose of `umask`?

10. What are the default permissions of a new file created by `touch`?

11. What is the function of `grep` and how is it used in log analysis?

12. How does `cut` differ from `awk`?

13. What is the use of the `find` command? Give syntax.

14. How does `sed` handle file editing?

15. What's the difference between `df -h` and `du -sh`?

16. What is the use of `ps`, `top`, and `htop`?

17. Explain how `cron` is used for scheduling in Linux.

18. What are environment variables and how are they set?

19. What does the pipe `|` operator do in Linux?

20. How is `tar` different from `zip`?

21. What’s the difference between `apt update`, `upgrade`, and `install`?

22. What is the use of `/etc/passwd` and `/etc/shadow`?

23. Explain process states in Linux (Running, Sleeping, Zombie).

24. What is the significance of inode in Linux?

25. Explain the concept of standard input, output, and error.

Section B: Practical Questions

26. Create a file named `sample.txt`, add three lines of text.

27. Create a directory structure: `/data/raw/2023`

28. Copy a file from `/home/user` to `/tmp`.

29. Rename a file using the `mv` command.

30. Display contents of a file with line numbers.

31. Append a line to a file without opening it.

32. Show the first 10 lines and last 5 lines of a file.

33. Extract 2nd column from a CSV using `cut`.

34. Sort a file alphabetically.

35. Sort a CSV file by column 3.

36. Replace all occurrences of the word "error" with "warning" in a log file.

37. Find all `.log` files under `/var/log`.

38. Show disk usage of each directory under `/home`.

39. Check available disk space and RAM in human-readable format.

40. Create a tar archive of a folder and extract it.

41. Compress a directory using `zip` and unzip it.

42. Add read and write permission to group and remove from others.

43. Change ownership of a file to user `dataeng` and group `engineers`.

44. Display only running processes of current user.

45. Count number of lines containing word `failed` in a file.

46. Create a cron job to run `backup.sh` every day at 5 AM.

47. Display current path and all environment variables.

48. Check permission details of all files in `/etc`.

49. Create a symbolic link to `/usr/bin/python3` as `python`.

50. Write a shell one-liner to print all `.csv` files with more than 10 lines.

51. Create a new user in Linux named `data_eng_user` with a home directory.

52. Add `data_eng_user` to the `sudoers` list and verify.

53. Create a directory `/opt/data_eng` and set **full permissions** only for the owner.

54. Change the owner of `/opt/data_eng` to `data_eng_user`.

55. Create an empty file named `students.txt` using three different methods.

56. Append the text `"Linux Assignment"` to `students.txt` without overwriting the file.

57. Display the **first 10 lines** of `/etc/passwd`.

58. Display the **last 15 lines** of `/var/log/syslog` (or any available log file).

59. Find the **current working directory** of the logged-in user.

60. List all files in `/etc` that start with the letter `h`.

61. Find all `.log` files inside `/var/log` recursively.

62. Display the **number of lines, words, and characters** in `/etc/passwd`.

63. Replace all occurrences of the word `root` with `admin` in a test file using `sed`.

64. Create a **symbolic link** for `/etc/passwd` inside `/tmp` directory.

65. Create a **hard link** for `/etc/passwd` inside `/tmp` directory.

66. Compress a file named `students.txt` using `gzip` and then decompress it.

67. Create a tar archive `backup.tar.gz` of the `/etc` directory.

68. Check the available **disk space** on the system.

69. Check the **memory usage** of the system.

70. Display all **running processes** and sort them by memory usage.

71. Kill a process by its **PID**.

72. Create a cron job to run `echo "Hello Data Engineering"` every 5 minutes.

73. Create a file named `file1.txt` and `file2.txt` and **merge** them into `merged.txt`.

74. Sort the contents of `merged.txt` in **descending order**.

75. Display only the **unique lines** from `merged.txt`.