Day-4 Bash Automation Assignments

File Backup Script

Scenario:
You are asked to create a script that protects important files by making backups. The script should accept a filename as input, generate a copy of the file with a `.bak` extension and a timestamp (so multiple backups don't overwrite each other), and log the backup activity to a central logfile.

Log Cleaner

Scenario:
Your system generates large `.log` files in `/var/tmp/`. Left unmanaged, these logs consume disk space and affect performance. You need to write a script that automatically finds and deletes `.log` files older than 7 days, ensuring no newer logs are touched.

Simple Monitoring

Scenario:
Operations teams need lightweight monitoring when resource-intensive tools are not available. Your script should check system CPU load from the `uptime` command every time it runs. If the load average is above 2.0, it should record an alert into a logfile with a timestamp.

Bulk User Creation

Scenario:
Your company is onboarding 20 new developers. You need to provision accounts `dev1` to `dev20`, add them to a `dev` group, and assign random passwords. The process must scale using automation rather than manual `useradd`. Passwords should be stored in a secure way (e.g., file with restricted permissions).

Automated Archiver

Scenario:
A data folder at `/srv/data/` must be backed up daily. Your script should compress the directory into a `.tar.gz` file named with the current date and move it to `/backups/`. Old archives older than 14 days must be deleted to prevent storage bloat.

Service Health Checker

Scenario:
Critical services like `sshd` (remote access) and `nginx` (web server) must always be available. Write a script that checks if these services are running. If a service is down, the script should restart it and log the recovery action with a timestamp.

Secret Scanner

Scenario:
Your company's source code must be checked for accidentally exposed secrets (like AWS keys). Build a script that scans through code repositories for sensitive patterns such as `AWS_SECRET_KEY`. The results must not print secrets on screen but instead be stored in a secure logfile with restricted access.