1- How do Linux file permissions (r, w, x) work for files vs directories? Give an example using ls -l.

- Linux permissions are represented by three characters: r (read), w (write), x (execute). They apply separately to: Owner (user who owns the file), Group (users in the file's group), and Others (all other users).
- For Files, r is read the file contents, w is to modify or delete the file, and x is to execute the file as a program/script. While for Directories, r is to list directory contents (ls), w is to create, delete, or rename files inside the directory, and x is to "enter" the directory (cd) and access files if you know their names.
- Example:

  ```
  "ls -l
  -rwxr--r-- 1 nada users  523 Aug 31  notes.sh
  drwxr-x--- 2 nada users 4096 Aug 31  projects"
  ```

  notes.sh is a regular file, Owner (nada) has rwx (read, write, execute), Group has r-- (read only), Others have r-- (read only).
  projects is a directory, Owner has rwx (can enter, list, and modify contents), and Group has r-x (can enter and list, but not modify).Others have --- (no access).

2- Explain octal notation for permissions and what the umask command does. Give one calculation example.

- In Octal notation permissionsare represented as numbers where the 3 positions of rwx represented as binary form, then they get transformed to decimal form by their values. So, rwx = 4+2+1 = 7, rw- = 4+2 = 6, r-- = 4 = 4.
- Umask command is simply a mask applied to the default number that represents the default rwx state of a new file and directory, which is applied by subtracting some specified number form the default number. So, If umask = 022, For files: 666 - 022 = 644 which is rw-r--r--, For directories: 777 - 022 = 755 which is rwxr-xr-x.

3- What is the difference between the root user and a normal user? Why is root considered dangerous?

- The root user is superuser in Linux, which has UID = 0 and full access to the system who can read/write/execute any file, manage users, install/remove software, modify kernel settings. It's dangerous as the root bypasses all security checks, so one wrong command (e.g., rm -rf /) can delete the entire system.

- Normal users have restricted privileges, can only access own files (or those shared with proper permissions), and it can’t change system-wide settings without using sudo.