Exercises on Permissions and Ownership

1. Explore and understand Permissions

1. Run:

   ```
   ls -l /etc/passwd
   ```

   What do the first 10 characters mean? Try to identify whether it's a file or directory, and which users/groups have which access.

2. Use another example:

   ```
   touch testfile && ls -l testfile
   ```

   Can you identify who owns the file and which group it belongs to?

Explain what each column in the `ls -l` output represents.

2. Changing Permissions with `chmod`

Goal: Practice symbolic and numeric modes.

1. Create a file:

   ```
   echo "Example text" > perms.txt
   ```

2. Give the owner execute permission (symbolic mode):

   ```
   chmod u+x perms.txt
   ```

Confirm with `ls -l perms.txt`.

3. Now use **numeric mode** to set permissions `rw-r--r--` (644):

   ```
   chmod 644 perms.txt
   ```

What's the difference between symbolic (`u+x`) and numeric (`755`) notation?

3. Change Ownership and Groups

Goal: Use `chown` and `chgrp` safely.

Suppose your system has users `student1` and a group `class`.

1. Change ownership:

```
sudo chown student1 perms.txt
```

2. Change the group:

```
sudo chgrp class perms.txt
```

3. Apply ownership changes recursively to a directory:

```
sudo chown -R student1:class myfolder/
```

Why can only `root` typically use `chown`?

4. Understanding `umask`

Goal: Control default file permissions.

1. View your current `umask`:

```
umask
```

2. Predict what permissions will be set when you create a new file with this `umask`.
3. Test your prediction:

```
touch my_new.txt && ls -l my_new.txt
```

If your `umask` is `022`, what are the default permissions for a new file and for a directory?

5. Special Modes: SUID, SGID, Sticky Bit

Goal: Understand and apply security bits.

1. Changing the **SUID** bit. What are the permissions of the file `/usr/bin/passwd`?

   Remove the sticky bit of `/usr/bin/passwd`.

   ```
   sudo chmod u-s /usr/bin/passwd
   ```

   As rocky, try to change your password. Explain why this fails.

   Restore the suid bit on /usr/bin/passwd.

   ```
   sudo chmod u+s /usr/bin/passwd
   ```

   Then inspect with `ls -l` — what letter replaces the `x` in the owner field?

2. Set the **SGID** bit on a directory:

   ```
   mkdir projects
   chmod g+s projects
   ```

3. Set the **Sticky bit** on a shared directory (e.g., `/tmp`):

   ```
   chmod +t projects
   ```

Which of these ensures only the file's owner can delete it from a shared directory?

6. Mini Review

Try to answer these before checking your notes:

- What numeric permission corresponds to `rwxr-xr--`?
- What does the `t` at the end of a permission string mean?
- How does `umask` 027 affect new directory permissions?