4 = r = read = View contents
2 = w = write = Modify contents
1 = x = execute = Execute Contents

Octal (Base-8) uses 0-7 for numbers.

4 = ?
5 = ?
7 = ?
1 = ?
6 = ?
System Administration

Directory Modes (Permissions)

4 = r = read = View list of directory contents
2 = w = write = Modify list of directory contents
1 = x = execute = Enter directory = cd to directory

some octal codes:
7 = ?
5 = ?
4 = ?
6 = ?
2 = ?
1 = ?
3 layers of modes:

User (UID) = mode applied to processes with matching UID.

Group (GID) = mode applied to processes with matching GID.

Other (world) = mode applied to all processes.

Each has the 3-bits: rwx.

If a process wants to read, it will be allowed if UID matches and user-read is set, or if GID matches and group-read is set, or if other-read is set. Similar rules apply to write and execute permissions.
3 special bits: In addition to user/group/other bits.

4 = setuid bit = s
2 = setgid bit = s
1 = sticky bit = t

File:

setuid = when program is run, the process will have the UID of the file, instead of inheriting from parent process.
setgid = similar, but for GID.
sticky = No effect.

Directories:

setuid = No effect
setgid = New contents of directory will have GID of directory instead of inheriting from creating process.
sticky = only root and file owners can delete and rename files in directory.
what is 2750?
6755?
1777?
0644?
-r-sr-sr-x?
-rwsr-Sr--?
-rwx rwx rw-t?
Default Permissions for new files:

0666 = ?

Default for new directories:

0777 = ?

Any problems with this?
Each process has a mask that defines permission bits that should be removed from newly created files and directories.

Some typical settings include:

0002 = ?

0027 = ?

This umask is inherited from parent process. Processes can change their umask with a system call. The shell usually gets the default umask from a shell initialization file.
Umask questions:

share with group but not world:

no write for anyone:

allow only execute for user and group, read for world: