Permissions

1. Dans le répertoire src, créez les fichiers vides droits-octal et droits-symb dont les droits sont par défaut fixés à rw-r-- r--

```
touch {droits-octal,droits-symb}
```

2. Avec la commande chmod, modifiez les droits de ces deux fichiers en parallèle en utilisant les valeurs en octal sur le fichier droits-octal et les lettres et symboles sur le fichier droits-symb de sorte qu'ils prennent successivement les valeurs ci-dessous :

```
# rw- -w- ---
chmod 620 droits-octal
chmod u=rw,g=w,o= droits-symb
# rwx -wx --x
chmod 731 droits-octal
chmod u=rwx,g=wx,o=x droits-symb
# r-x -w- --x
chmod 521 droits-octal
chmod u=rx,g=w,o=x droits-symb
# --x --x r--
chmod 114 droits-octal
chmod ug=x,o=r droits-symb
```

3. Faites une copie du fichier droits-octal appelée droits-octal-copie.

```
cp droits-octal droits-octal-copie
# Output
"cp: cannot open 'droits-octal' for reading: Permission denied"
```

4. Expliquez le problème rencontré et résolvez-le à l'aide de la commande chmod.

```
# L'utilisateur n'as pas les permissions de lecture et ne peut donc pas copier le fichier
chmod u+r droits-octal
cp droits-octal droits-octal-copie
```

5. Que constatez-vous au niveau des droits de la copie ?

    i. Elle hérite les permissions

6. Sur le répertoire src qui vous appartient, appliquez successivement les combinaisons de droits d'accès listées ci-dessous et à chaque fois essayez de :

    - Rentrer dans le répertoire src (cd ./src)
    - Lister le contenu détaillé du répertoire src en affichant les informations sur les fichiers (ls –l ./src)
    - Modifier (ouvrir, modifier puis enregistrer) avec nano le fichier src/prog2.c (nano src/prog2.c)
    - Supprimer (puis recréer si besoin) le fichier src/prog2.c (rm src/prog2.c)

```
# Droits cd ./src ls –l ./src nano src/prog2.c rm src/prog2.c
# 600 (rw-)
chmod -R 600 ./src
cd ./src
```

```
# out
"cd: permission denied: src"
ls -l ./src
# out
"ls: cannot access './src/droits-octal-copie': Permission denied
ls: cannot access './src/prog2.c': Permission denied
ls: cannot access './src/droits-octal': Permission denied
ls: cannot access './src/droits-symb': Permission denied
total 0
-????????? ? ? ? ?            ? droits-octal
-????????? ? ? ? ?            ? droits-octal-copie
-????????? ? ? ? ?            ? droits-symb
-????????? ? ? ? ?            ? prog2.c"
vim src/prog2.c
# out
"Permission Denied"
rm src/prog2.c
# out
"rm: cannot remove 'src/prog2.c': Permission denied"
# 500 (r-x)
chmod -R 500 ./src
cd src
# No issue
ls -l ./src
# out
"total 0
-r-x------ 1 mon mon 0 May 10 08:57 droits-octal
-r-x------ 1 mon mon 0 May 10 09:14 droits-octal-copie
-r-x------ 1 mon mon 0 May 10 08:57 droits-symb
-r-x------ 1 mon mon 0 May  4 11:27 prog2.c"
vim src/prog2.c
# readonly, cannot save
# can be bypassed with :wq! if the user has the sudo permissions
rm src/prog2.c
# out
"rm: remove write-protected regular file 'src/prog2.c'? y
rm: cannot remove 'src/prog2.c': Permission denied"
# 100 (--x)
chmod -R 100 ./src
# errors for the recursive part
# using sudo, we can see that the permissions on the children didn't change
cd src
# No issue
ls -l
# out
"ls: cannot open directory '.': Permission denied"
vim src/prog2.c
# File is readonly, can still be bypassed with :w!
rm src/prog2.c
"rm: remove write-protected regular file 'src/prog2.c'? y
rm: cannot remove 'src/prog2.c': Permission denied"
# 300 (-wx)
chmod -R 300 ./src
# directory permissions changed but children permission stayed the same since chmod cannot read files
ls -l src
# out
"ls: cannot open directory 'src': Permission denied"
vim src/prog2.c
# same as before, file permissions didn't change
rm src/prog2.c
# able to delete file since the user has write permissions on the folder
# 700 (rwx)
# Recreate prog2.c before changing permissions
# Otherwise change permissions, create the file and set its permissions
sudo touch ./src/prog2.c
sudo chown mon:mon src/prog2.c
chmod -R 700 ./src
cd src
cd ..
```

```
ls -l ./src
# out
"total 0
-rwx------ 1 mon mon 0 May 10 08:57 droits-octal
-rwx------ 1 mon mon 0 May 10 09:14 droits-octal-copie
-rwx------ 1 mon mon 0 May 10 08:57 droits-symb
-rwx------ 1 mon mon 0 May 10 10:29 prog2.c"
vim src/prog2.c
rm src/prog2.c
```

7. A partir de ces résultats, retrouvez les règles générales sur l'accès aux répertoires.

```
r = list files in the directory
w = remove or add file to the directory
x = enter directory/interact with files inside of the directory
```

8. Connectez-vous en utilisateur root avec la commande su puis attribuez-vous (à root) le fichier prog3.for en utilisant la commande chown.

```
sudo chown root prog3.for
```

9. Toujours en root, changez le groupe propriétaire de ce fichier pour qu'il appartienne au groupe mail.

```
chown :mail prog3.for
```

10. Comment combiner les 2 dernières commandes en une seule ?

```
chown root:mail prog3.c
```

11. Toujours connecté en utilisateur root, je désire :

    - Attribuer la sous-arborescence du répertoire src à l'utilisateur nobody et au groupe users

      ```
      chown -R nobody:users src
      ```

      i. Affichez la valeur par défaut du umask pour l'utilisateur lambda.

      ```
      su mon
      umask
      # out
      022
      ```

12. Passez en root et affichez la valeur de son umask.

```
su root
umask
# out
0022
```

13. Quelle raison permet d'expliquer la différence entre ces 2 valeurs ?

    - le premier byte definis setuid,setgid et un sticky bit qui est defini pour root mais pas pour l'utilisateur lambda

14. Quelle unique valeur donner à la commande umask pour obtenir les droits ci-dessous à la création de (plusieurs réponses possibles!):

```
# nouveaux fichiers : rw- r-- ---
# nouveaux répertoires : rwx r-x ---
umask 027
# Only new directories are affected
```