

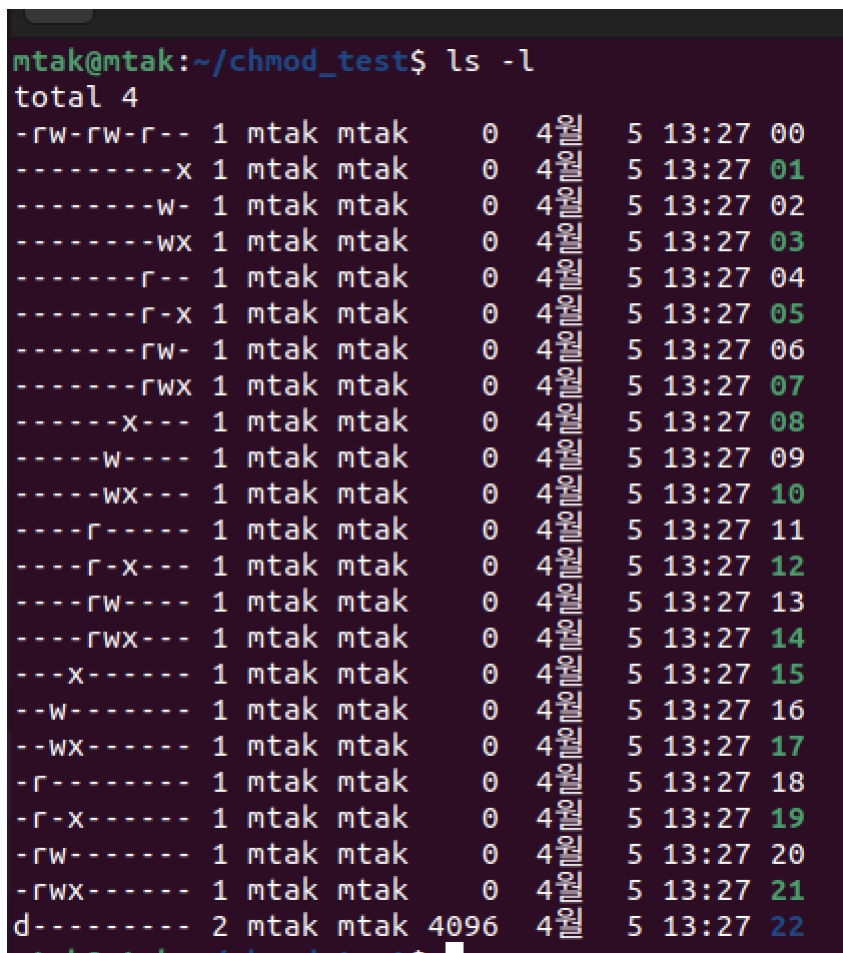
Homework #3

1. Do "mkdir ~/chmod_test", then do "cd ~/chmod_test", then do the following commands one by one.

```
for t in {0..21}; do if [ $t -lt 10 ]; then touch 0"$t"; else touch $t; fi
done
i=1; for t in {1..7}; do if [ $t -lt 10 ]; then t=0"$t"; fi; chmod 000"$i"
$t; i=$((i+1)); done
i=1; for t in {8..14}; do if [ $t -lt 10 ]; then t=0"$t"; fi; chmod
00"$i"0 $t; i=$((i+1)); done
i=1; for t in {15..21}; do chmod 0"$i"00 $t; i=$((i+1)); done
mkdir 22
chmod 000 22
```

Then, do "ls -l".

- Take a screenshot



```
mtak@mtak:~/chmod_test$ ls -l
total 4
-rw-rw-r-- 1 mtak mtak 0 4월 5 13:27 00
-----x 1 mtak mtak 0 4월 5 13:27 01
-----w- 1 mtak mtak 0 4월 5 13:27 02
-----wx 1 mtak mtak 0 4월 5 13:27 03
-----r-- 1 mtak mtak 0 4월 5 13:27 04
-----r-x 1 mtak mtak 0 4월 5 13:27 05
-----rw- 1 mtak mtak 0 4월 5 13:27 06
-----rwx 1 mtak mtak 0 4월 5 13:27 07
-----x-- 1 mtak mtak 0 4월 5 13:27 08
-----w--- 1 mtak mtak 0 4월 5 13:27 09
-----wx-- 1 mtak mtak 0 4월 5 13:27 10
----r----- 1 mtak mtak 0 4월 5 13:27 11
----r-x--- 1 mtak mtak 0 4월 5 13:27 12
----rw---- 1 mtak mtak 0 4월 5 13:27 13
----rwx--- 1 mtak mtak 0 4월 5 13:27 14
---x----- 1 mtak mtak 0 4월 5 13:27 15
--w----- 1 mtak mtak 0 4월 5 13:27 16
--wx----- 1 mtak mtak 0 4월 5 13:27 17
-r----- 1 mtak mtak 0 4월 5 13:27 18
-r-x----- 1 mtak mtak 0 4월 5 13:27 19
-rw----- 1 mtak mtak 0 4월 5 13:27 20
-rwx----- 1 mtak mtak 0 4월 5 13:27 21
d----- 2 mtak mtak 4096 4월 5 13:27 22
```

- List files which can be read by file owner 00, 18, 19, 20, 21
- List files which can be executed by file group members 08, 10, 12, 14,
- List files which can be written by others. 02, 03, 06, 07

2. Do the following commands one by one.

```
i=1; for t in {1..7}; do if [ $t -lt 10 ]; then t=0"$t"; fi; chmod 700"$i"
$t; i=$((i+1)); done
i=1; for t in {8..14}; do if [ $t -lt 10 ]; then t=0"$t"; fi; chmod
70"$i"0 $t; i=$((i+1)); done
i=1; for t in {15..21}; do chmod 7"$i"00 $t; i=$((i+1)); done
```

Then do "ls -l".

```
mtak@mtak:~/chmod_test$ ls -l
total 4
-rw-rw-r-- 1 mtak mtak 0 4월 5 13:27 00
---S---S---t 1 mtak mtak 0 4월 5 13:27 01
---S---S-wT 1 mtak mtak 0 4월 5 13:27 02
---S---S-wt 1 mtak mtak 0 4월 5 13:27 03
---S---Sr-T 1 mtak mtak 0 4월 5 13:27 04
---S---Sr-t 1 mtak mtak 0 4월 5 13:27 05
---S---SrWT 1 mtak mtak 0 4월 5 13:27 06
---S---Srwt 1 mtak mtak 0 4월 5 13:27 07
---S---s--T 1 mtak mtak 0 4월 5 13:27 08
---S-wS--T 1 mtak mtak 0 4월 5 13:27 09
---S-ws--T 1 mtak mtak 0 4월 5 13:27 10
---Sr-S--T 1 mtak mtak 0 4월 5 13:27 11
---Sr-s--T 1 mtak mtak 0 4월 5 13:27 12
---SrwS--T 1 mtak mtak 0 4월 5 13:27 13
---Srws--T 1 mtak mtak 0 4월 5 13:27 14
---s--S--T 1 mtak mtak 0 4월 5 13:27 15
---wS--S--T 1 mtak mtak 0 4월 5 13:27 16
---ws--S--T 1 mtak mtak 0 4월 5 13:27 17
-r-S--S--T 1 mtak mtak 0 4월 5 13:27 18
-r-s--S--T 1 mtak mtak 0 4월 5 13:27 19
-rwS--S--T 1 mtak mtak 0 4월 5 13:27 20
-rws--S--T 1 mtak mtak 0 4월 5 13:27 21
d----- 2 mtak mtak 4096 4월 5 13:27 22
```

- Take a screenshot
- Discuss the difference between files with "S" or "T" and files "s" or "t". case-insensitive s, t 는 모두 special permission이다. special permission은 execute자리들을 무단 점거한다. 따라서 원래 파일 권한 중 execute 영역이 activated이었는데 구분해줄 필요가 생겼기에 이를 대소문자로 구분한다. in detail, s, t 가 소문자라면 각각의 special permission 를 나타냄과 동시에 execute 권한이 있음을 나타낸다. 반대로 대문자라면 execute 권한이 없이 special permission 만 있다는 의미이다. setuid는 8진수 (4000)이나 기호(u+s)로 설정해 줄 수 있고, setuid 비트가 설정된 파일은 실행 순간만 그 파일의 소유주 권한으로 실행된다. setgid는 8진수(200)이나 기호(g+s)로 설정해 줄 수 있고, 특히 디렉토리에 setgid 비트가 설정되었다면, 이후 새로운 파일이나 폴더가 생성될 때 그 그룹 소유권은 해당 디렉토리의 그룹 소유권을 따라간다.

3. Do "rm ??", then press "y" key whenever cursor waits for your input. Then, do "rmdir 22"

```
d----- 2 mtak mtak 4096 4월 5 13:27 22
mtak@mtak:~/chmod_test$ rm ??
rm: remove write-protected regular empty file '01'? y
rm: remove write-protected regular empty file '02'? y
rm: remove write-protected regular empty file '03'? y
rm: remove write-protected regular empty file '04'? y
rm: remove write-protected regular empty file '05'? y
rm: remove write-protected regular empty file '06'? y
rm: remove write-protected regular empty file '07'? yy
rm: remove write-protected regular empty file '08'? y
rm: remove write-protected regular empty file '09'? y
rm: remove write-protected regular empty file '10'? y
rm: remove write-protected regular empty file '11'? y
rm: remove write-protected regular empty file '12'? y
rm: remove write-protected regular empty file '13'? y
rm: remove write-protected regular empty file '14'? y
rm: remove write-protected regular empty file '15'? y
rm: remove write-protected regular empty file '18'? y
rm: remove write-protected regular empty file '19'? y
rm: cannot remove '22': Is a directory
mtak@mtak:~/chmod_test$ rmdir 22
```

- Take a screenshot
- List the files which remove without the interactive question. And describe the common characteristics of these files. 00,16,17,20,21 이 파일들은 모두 owner(user)에게 쓰기 권한이 있었다.

4. Do the following commands one by one.

```
touch a b c d e f g h i
chmod = *
chmod 714 a
chmod u+rw b
chmod u+xs,g+r,o= c
chmod ug+rw,o+r d
chmod u+rw,g+rws,o= e
chmod ugo+x f
chmod 755 g
chmod 7744 h
chmod u=,g=,o=t i
```

Then, do "ls -al"

```
mtak@mtak:~/chmod_test$ ls -al
total 8
drwxrwxr-x  2 mtak mtak 4096 4월 5 14:09 .
drwxr-x--- 16 mtak mtak 4096 4월 5 10:27 ..
-rwx--xr--  1 mtak mtak    0 4월 5 14:09 a
-rw-----  1 mtak mtak    0 4월 5 14:09 b
---sr----- 1 mtak mtak    0 4월 5 14:09 c
-rw-rw-r--  1 mtak mtak    0 4월 5 14:09 d
-rw-rws---  1 mtak mtak    0 4월 5 14:09 e
---x--x--x  1 mtak mtak    0 4월 5 14:09 f
-rwxr-xr-x  1 mtak mtak    0 4월 5 14:09 g
-rwsr-Sr-T  1 mtak mtak    0 4월 5 14:09 h
-----T    1 mtak mtak    0 4월 5 14:09 i
```

- Take a screenshot

5. do the following commands.

```
for t in {0..7}; do if [ $t -lt 10 ]; then mkdir 0"$t"; else mkdir $t; fi
done
i=1; for t in {1..7}; do if [ $t -lt 10 ]; then t=0"$t"; fi; chmod 0"$i"00
$t; i=$((i+1)); done
ls -l
```

- Take a screenshot

```
mtak@mtak:~/chmod_test$ for t in {0..7}; do if [ $t -lt 10 ]; then mkdir 0"$t"; else mkdir $
t; fi done
i=1; for t in {1..7}; do if [ $t -lt 10 ]; then t=0"$t"; fi; chmod 0"$i"00 $t; i=$((i+1));
done
ls -l
total 32
drwxrwxr-x 2 mtak mtak 4096 4월 5 14:39 00
d--x----- 2 mtak mtak 4096 4월 5 14:39 01
d-w----- 2 mtak mtak 4096 4월 5 14:39 02
d-wx----- 2 mtak mtak 4096 4월 5 14:39 03
dr----- 2 mtak mtak 4096 4월 5 14:39 04
dr-x----- 2 mtak mtak 4096 4월 5 14:39 05
drw----- 2 mtak mtak 4096 4월 5 14:39 06
drwx----- 2 mtak mtak 4096 4월 5 14:39 07
-rwx--xr-- 1 mtak mtak    0 4월 5 14:09 a
-rw----- 1 mtak mtak    0 4월 5 14:09 b
---sr----- 1 mtak mtak    0 4월 5 14:09 c
-rw-rw-r-- 1 mtak mtak    0 4월 5 14:09 d
-rw-rws--- 1 mtak mtak    0 4월 5 14:09 e
---x--x--x 1 mtak mtak    0 4월 5 14:09 f
-rwxr-xr-x 1 mtak mtak    0 4월 5 14:09 g
-rwsr-Sr-T 1 mtak mtak    0 4월 5 14:09 h
-----T    1 mtak mtak    0 4월 5 14:09 i
```

Then, do the following command

```
i=1; for t in {1..7}; do if [ $t -lt 10 ]; then t=0"$t"; fi; echo "touch
$t/a"; touch $t/a; i=$((i+1)); done
```

- Take a screenshot

```
mtak@mtak:~/chmod_test$ i=1; for t in {1..7}; do if [ $t -lt 10 ]; then t=0"$t"; fi; echo "touch $t/a"; touch $t/a; i=$((i+1)); done
touch 01/a
touch: cannot touch '01/a': Permission denied
touch 02/a
touch: cannot touch '02/a': Permission denied
touch 03/a
touch 04/a
touch: cannot touch '04/a': Permission denied
touch 05/a
touch: cannot touch '05/a': Permission denied
touch 06/a
touch: cannot touch '06/a': Permission denied
touch 07/a
mtak@mtak:~/chmod_test$
```

- List directories which contain an empty file "a". And describe why they can generate the file.

03 과 07 directory 에만 a 파일이 생성되었는데, 03 과 07 에는 user 권한으로 쓰기 권한과 실행 권한을 모두 가지고 있어서 (x)로 directory에 접근한 후에 파일을 생성할 수(w) 있었다.

Then do the following command

```
i=1; for t in {1..7}; do if [ $t -lt 10 ]; then t=0"$t"; fi; echo "ls $t";
ls -al $t; i=$((i+1)); done
```

- Take a screenshot

```
mtak@mtak:~/chmod_test$ i=1; for t in {1..7}; do if [ $t -lt 10 ]; then t=0"$t"; fi; echo "ls $t"; ls -al $t; i=$((i+1)); done
ls 01
ls: cannot open directory '01': Permission denied
ls 02
ls: cannot open directory '02': Permission denied
ls 03
ls: cannot open directory '03': Permission denied
ls 04
ls: cannot access '04/.': Permission denied
ls: cannot access '04/..': Permission denied
total 0
d???????? ? ? ? ? ? ? ? ?
d???????? ? ? ? ? ? ? ? ?
ls 05
total 8
dr-x----- 2 mtak mtak 4096 4월 5 14:39 .
drwxrwxr-x 10 mtak mtak 4096 4월 5 14:39 ..
ls 06
ls: cannot access '06/.': Permission denied
ls: cannot access '06/..': Permission denied
total 0
d???????? ? ? ? ? ? ? ? ?
d???????? ? ? ? ? ? ? ? ?
ls 07
total 8
drwx----- 2 mtak mtak 4096 4월 5 14:40 .
drwxrwxr-x 10 mtak mtak 4096 4월 5 14:39 ..
-rw-rw-r-- 1 mtak mtak 0 4월 5 14:40 a
```

- List directories which print some results for "ls" command. Why?

04,05,06,07 directory. 이 디렉토리들은 user 권한에 읽기권한(r)이 있기 때문에 ls 명령어의 결과를 출력한다.

- Find a directory which presents the correct details of its sub files. Why? 05,07 directory. 이 디렉토리들은 user 권한에 읽기 권한(r)과 뿐만 아니라 실행권한(x)도 있기에 sub file의 정보를 나타낼 수 있다.

PROBLEM !!

How to delete the directory "03" ?

일단 디렉토리 03 에는 a 파일이 있어 빈 디렉토리가 아니기 때문에 rmdir 03 으로는 지울수 없다. 같은 이유로 디렉토리 03 에 read 권한이 없기 때문에 디렉토리에 있는 파일 및 서브 디렉토리를 읽을 수 없어 rm -r 03으로도 지울 수 없다. 고로 chmod u+r로 내부 내용물을 읽을 수 있는 권한을 추가해 준 다음, rm -r 03을 하면 03폴더가 통째로 사라진다.

6. Do "mkdir ~/homework3", then do "cd ~/homework3", then do "su", then do "cp /bin/bash backbash", then do "chmod u+s backbash", then do "ls -l backbash", then do "touch c", then do "chmod 600 c", then do "exit", then do "cat c", then do "id", then do "./backbash -p", then do "cat c", then do "id", then do "exit". Then do "rm -f backbash c"

- Take a screenshot

```
mtak@mtak:~$ mkdir homework3
mtak@mtak:~$ cd homework3/
mtak@mtak:~/homework3$ ls
mtak@mtak:~/homework3$ su
Password:
root@mtak:/home/mtak/homework3# cp /bin/bash/backbash
cp: missing destination file operand after '/bin/bash/backbash'
Try 'cp --help' for more information.
root@mtak:/home/mtak/homework3# cp /bin/bash backbash
root@mtak:/home/mtak/homework3# chmod u+s backbash
root@mtak:/home/mtak/homework3# ls -l backbash
-rwsr-xr-x 1 root root 1396520  4월  5 15:21 backbash
root@mtak:/home/mtak/homework3# touch c
root@mtak:/home/mtak/homework3# chmod 600 c
root@mtak:/home/mtak/homework3# exit
exit
mtak@mtak:~/homework3$ cat c
cat: c: Permission denied
mtak@mtak:~/homework3$ id
uid=1000(mtak) gid=1000(mtak) groups=1000(mtak),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),122(lpadmin),135(lxd),136(sambashare),1003(family)
mtak@mtak:~/homework3$ ./backbash -p
backbash-5.1# cat c
backbash-5.1# id
uid=1000(mtak) gid=1000(mtak) euid=0(root) groups=1000(mtak),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),122(lpadmin),135(lxd),136(sambashare),1003(family)
backbash-5.1# exit
exit
mtak@mtak:~/homework3$ rm -f backbash c
mtak@mtak:~/homework3$
```

- Explain the difference between the first "cat" result and the second "cat" result. 첫번째는 mtak에서 cat 이 실행되어 c 에 대한 권한이 없었기에 명령이 거부되었는데, 두번째 cat 은 backbash -p로 root권한을 얻어 실행한 거라, 명령이 받아들여졌다.

- Describe the difference between the first "id" result and the second "id" result. 첫번째는 euid(유효 사용자)가 mtak인 상태로 id 명령어를 실행시켰는데, ./backbash -p 명령으로 인해 euid가 root인 상태로 실행시켰다.

7. Do "su – peterpan" (user peterpan should exist), then do "clear", then do "mkdir proj1", then do "mkdir proj1/sub1", then do "ls -l proj1", then do "chgrp defender proj1", then do "chmod g+s proj1", then do "mkdir proj1/sub2", then do "touch proj1/a", then do "ls -l proj1".

- Take a screenshot

```
peterpan@mtak:~$ mkdir proj1
peterpan@mtak:~$ mkdir proj1/sub1
peterpan@mtak:~$ ls -l proj1/
total 4
drwxrwxr-x 2 peterpan peterpan 4096  4월  6 09:20 sub1
peterpan@mtak:~$ chgrp defender
.bash_logout .bashrc      .profile      proj1/
peterpan@mtak:~$ chgrp defender proj1/
peterpan@mtak:~$ chmod g+s proj1/
peterpan@mtak:~$ mkdir proj1/sub2
peterpan@mtak:~$ touch proj1/a
peterpan@mtak:~$ ls -l proj1/
total 8
-rw-rw-r-- 1 peterpan defender    0  4월  6 09:21 a
drwxrwxr-x 2 peterpan peterpan 4096  4월  6 09:20 sub1
drwxrwsr-x 2 peterpan defender 4096  4월  6 09:21 sub2
peterpan@mtak:~$
```

- What is difference between the "sub1" directory and the "sub2" directory? Why?

sub1 디렉토리의 그룹 소유권은 peterpan 이고 sub2 디렉토리의 그룹소유권은 defender이다. 그룹 소유권이 처음 그룹소유권인 peterpan일 때 sub1을 만들어서 sub1의 그룹소유권은 peterpan이다. 이후 chgrp defender proj1 으로 인해 그룹 소유권을 defender 로 변경한 후, setgid를 지정해 주었기 때문에 이후 디렉토리에서 생성되는 모든 것들의 그룹 소유권은 defender이다. 고로 이후에 sub2 를 생성하였으므로 sub2 의 그룹소유권은 defender이다.

- What is the ownership of the file proj1/a. Why? 유저소유권은 peterpan 이고, 그룹소유권은 defender 인데, 앞서 설명한 바와 같이 디렉토리의 그룹 소유권이 defender이기 때문에 그 안에 만들어진 파일 a도 그룹 소유권이 defender이다.

8. Do "clear", then do "mkdir shared", then do "chmod 777 shared", then do "mkdir shared_t", then do "chmod 1777 shared_t", then do "touch shared/a", then do "touch shared_t/a", then do "cd shared", then do "su

hook" (user hook should exist), then do "rm a", then do "exit", then do

"cd ../shared_t", then do "su hook" (user hook should exist), then do "rm a", then do "exit", then do "exit".

- Take a screenshot

```
peterpan@mtak:~$ mkdir shared; chmod 777 shared; mkdir shared_t; chmod 1777 shared_t;
peterpan@mtak:~$ touch shared/a; touch shared_t/a
peterpan@mtak:~$ cd shared
peterpan@mtak:~/shared$ su hook
Password:
hook@mtak:/home/peterpan/shared$ rm a
rm: remove write-protected regular empty file 'a'? y
hook@mtak:/home/peterpan/shared$ exit
exit
peterpan@mtak:~/shared$ cd ../shared_t/
peterpan@mtak:~/shared_t$ su hook
Password:
hook@mtak:/home/peterpan/shared_t$ rm a
rm: remove write-protected regular empty file 'a'? y
rm: cannot remove 'a': Operation not permitted
hook@mtak:/home/peterpan/shared_t$ exit
exit
peterpan@mtak:~/shared_t$ exit
exit
hook@mtak:/home/peterpan$
```

- Which file is removed? Why? shared/a 파일만이 삭제 되었는데, shared_t 에는 special permission 인 sticky 설정을 해놔서 hook 의 삭제 명령이 허용되지 않았다. shared/a 파일은 모든 사용자에게 권한을 부여했기 때문에 삭제가 되었다.

Problem

1. 유저 wendy의 home directory에 secret이라는 directory를 만든다. 이 directory의 내용은 peterpan과 wendy만이 볼수 있다. 이 directory내에 letter_to_peter 라는 파일을 만들고 그 안에 "Hi Peter, See you on Monday 😊" 라는 내용을 적어 넣는다. 이 파일은 peterpan과 wendy만 볼수

있게 해야 하며, peterpan은 이파일의 내용을 수정하지 못한다. 하지만 peterpan은 이 directory에 파일을 생성 할 수 있다.

- 수행해야 할 명령어들을 순서대로 나열하고 그 역할을 설명하시오.

```
#시작 유저 mtak 은 sudoer이다.
su wendy
cd ~
mkdir secret; sudo chgrp peterpan secret;
chmod 1770 ../secret
cd secret
echo "Hi Peter, See you on Monday :)" > letter_to_peter
chmod 604 letter_to_peter
```


1. wendy로 유저를 바꾼다.
 2. wendy의 홈으로 이동한다
 3. secret폴더를 만들고 그룹 소유권을 peterpan으로 변경한다.
 4. secret의 모드를 1770으로 수정하여 peterpan, wendy가 cd로 접근 가능하고, 파일/폴더를 생성 가능하게 만든다.
그리고 other의 디렉토리 내용을 삭제를 막아놓았다.
 5. secret으로 이동한다.
 6. letter_to_peter파일을 만든다.
 7. letter_to_peter의 모드를 604으로 만들어 peterpan이 파일을 삭제할(w)수 없게 만들었다.
- 결과의 확인을 위해 wendy의 home directory에서 "ls -l"을 수행한 결과와, secret directory에서 "ls -l"을 수행한 결과를 캡처한다.

```
wendy@mtak:~$ ls -l
total 4
drwxrwx--T 2 wendy peterpan 4096  4월  6 14:10 secret
wendy@mtak:~$
```

2. 정확한 접근권한과 소유권을 가지는 다음의 파일들과 디렉토리들을 생성하시오.

1. 이 작업을 위한 명령어를 순서대로 나열하시오.
2. 확인을 위해 "ls -al" 결과를 첨부하시오.

```
--wx-w-r-T 1 peterpan neverland 0 Mar 22 01:01 a
-rw-rw-r-- 1 wendy neverland 0 Mar 22 01:01 .a
--w-r-Sr-t 1 peterpan defender 0 Mar 22 01:01 b
---x--Sr-T 1 peterpan defender 0 Mar 22 01:02 c
---S--S-wT 1 wendy defender 0 Mar 22 01:02 d
--wx-wx-wx 1 wendy defender 0 Mar 22 01:02 e
-rw-r-xr-x 1 hook attacker 0 Mar 22 01:02 f
-rwxr-xr-- 1 hook attacker 0 Mar 22 01:02 g
-r-sr-xr-x 1 hook attacker 0 Mar 22 01:04 h
drwxr-xr-x 2 peterpan neverland 4096 Mar 22 01:05 sec
drwxr-xr-t 2 wendy neverland 4096 Mar 22 01:01 .sec
```

3. 유저 wendy의 home directory에 "test1" 파일을 만든다. 이 파일의 내용은 같은 그룹의 peterpan과 wendy만 보고 수정할 수 있게 해야 하며, 그외 사람들은 볼 수만 있게 한다.

- 수행해야 할 명령어들을 순서대로 나열하고 그 역할을 설명하시오.

유저 peterpan, wendy, hook, 그룹 neverland, defender, attacker 는 미리 생성되어 있었고, 시작 유저는 root 이다.

```
mkdir test;cd test
touch a .a b c d e f g h
mkdir sec .sec
chown peterpan:neverland a; chmod 1324 a
chown wendy:neverland .a; chmod 664 .a
chown peterpan:defender b; chmod 3245 b
```

```
chown peterpan:defender c; chmod 3104 c
chown wendy:defender d; chmod 7002 d
chown wendy:defender e; chmod 333 e
chown hook:attacker f; chmod 655 f
chown hook:attacker g; chmod 754 g
chown attacker:attacker h; chmod 4555 h
chown peterpan:neverland sec; chmod 755 sec
chown wendy:neverland .sec; chmod 1755 .sec
```

1. 파일을 만들 test폴더를 만들고 이동한다.
 2. 필요한 파일과 폴더를 만든다.
 3. chown으로 owner user, group을 설정해주고, chmod로 권한을 설정해준다.
- 결과의 확인을 위해 유저를 변경하며 다음의 명령어를 수행하고 그 결과를 확인하시오 .

```
sudo su wendy
echo The $USER can revise the test1 > test1;
cat test1;
exit
sudo su peterpan
echo The $USER can revise the test1 > test1
cat test1
exit
sudo su hook
echo The $USER can revise the test1 > test1
cat test1
exit
```

```
wendy@mtak:~/test$ echo The $USER can revise the test1 > test1;
cat test1;
The wendy can revise the test1
wendy@mtak:~/test$ exit
exit
root@mtak:/home/wendy/test# sudo su peterpan
peterpan@mtak:/home/wendy/test$ echo The $USER can revise the test1 > test1;
cat test1;
bash: test1: Permission denied
The wendy can revise the test1
peterpan@mtak:/home/wendy/test$ exit
exit
root@mtak:/home/wendy/test# sudo su hook
hook@mtak:/home/wendy/test$ echo The $USER can revise the test1 > test1;
cat test1;
bash: test1: Permission denied
The wendy can revise the test1
hook@mtak:/home/wendy/test$ exit
exit
root@mtak:/home/wendy/test#
```

4. Simply explain chmod command - Absolute Mode and Symbolic Mode.

접근 권한을 만들 때 8진수로 표현하면 절대 모드로 변경하는 것이고, r, w, x로 표현하면 상대 모드로 변경하는 것이다.