Exercises

Create a new directory called missing under /tmp.

side@ubuntu:~$ cd /tmp
side@ubuntu:/tmp$ mkdir missing

Look up the touch program. The man program is your friend.

side@ubuntu:/tmp$ man touch
side@ubuntu:/tmp$ cd missing
side@ubuntu:/tmp/missing$ touch semester
side@ubuntu:/tmp/missing$ ls -l
total 0
-rw-r--r-- 1 side side 0 May 2 23:28 semester

Write the following into that file, one line at a time:

side@ubuntu:/tmp/missing$ vim semester

```
#!/bin/sh
curl --head --silent https://missing.csail.mit.edu
```

side@ubuntu:/tmp/missing$ ls -l
total 4
-rw-r--r-- 1 side side 61 May 2 23:31 semester
side@ubuntu:/tmp/missing$

Try to execute the file, i.e. type the path to the script (./semester) into your shell and press enter. Understand why it doesn't work by consulting the output of ls (hint: look at the permission bits of the file).

side@ubuntu:/tmp/missing$ ./semester
bash: ./semester: Permission denied

Run the command by explicitly starting the sh interpreter, and giving it the file semester as the first argument, i.e. sh semester. Why does this work, while ./semester didn't?

```
side@ubuntu:/tmp/missing$ sh semester
HTTP/2 200
content-type: text/html; charset=utf-8
server: GitHub.com
last-modified: Sat, 25 Apr 2020 11:12:41 GMT
etag: "5ea41b29-1abd"
access-control-allow-origin: *
expires: Sun, 03 May 2020 05:41:41 GMT
cache-control: max-age=600
x-proxy-cache: MISS
x-github-request-id: E6BE:7468:258856:2A45C3:5EAE573D
accept-ranges: bytes
date: Sun, 03 May 2020 06:33:41 GMT
via: 1.1 varnish
age: 0
x-served-by: cache-bom18220-BOM
x-cache: MISS
x-cache-hits: 0
x-timer: S1588487622.530960,VS0,VE192
vary: Accept-Encoding
x-fastly-request-id: 3e1589843973b632f5402277757c3d264342bacf
content-length: 6845
```

Look up the chmod program (e.g. use man chmod).

```
side@ubuntu:/tmp/missing$ man chmod


CHMOD(1)                    User Commands                    CHMOD(1)

NAME
       chmod - change file mode bits

SYNOPSIS
       chmod [OPTION]... MODE[,MODE]... FILE...
       chmod [OPTION]... OCTAL-MODE FILE...
       chmod [OPTION]... --reference=RFILE FILE...

DESCRIPTION
       This  manual  page documents the GNU version of chmod.  chmod changes the file mode
       bits of each given file according to mode, which can be either a symbolic represen-
       tation  of changes to make, or an octal number representing the bit pattern for the
       new mode bits.

       The format of a symbolic mode  is  [ugoa...][[-+=][perms...]...],  where  perms  is
```

either zero or more letters from the set rwxXst, or a single letter from the set ugo. Multiple symbolic modes can be given, separated by commas.

A combination of the letters ugoa controls which users' access to the file will be changed: the user who owns it (u), other users in the file's group (g), other users not in the file's group (o), or all users (a). If none of these are given, the effect is as if (a) were given, but bits that are set in the umask are not affected.

The operator + causes the selected file mode bits to be added to the existing file mode bits of each file; - causes them to be removed; and = causes them to be added and causes unmentioned bits to be removed except that a directory's unmentioned set user and group ID bits are not affected.

The letters rwxXst select file mode bits for the affected users: read (r), write (w), execute (or search for directories) (x), execute/search only if the file is a directory or already has execute permission for some user (X), set user or group ID on execution (s), restricted deletion flag or sticky bit (t). Instead of one or more of these letters, you can specify exactly one of the letters ugo: the permissions granted to the user who owns the file (u), the permissions granted to other users who are members of the file's group (g), and the permissions granted to users that are in neither of the two preceding categories (o).

A numeric mode is from one to four octal digits (0-7), derived by adding up the bits with values 4, 2, and 1. Omitted digits are assumed to be leading zeros. The first digit selects the set user ID (4) and set group ID (2) and restricted deletion or sticky (1) attributes. The second digit selects permissions for the user who owns the file: read (4), write (2), and execute (1); the third selects permissions for other users in the file's group, with the same values; and the fourth for other users not in the file's group, with the same values.

chmod never changes the permissions of symbolic links; the chmod system call cannot change their permissions. This is not a problem since the permissions of symbolic links are never used. However, for each symbolic link listed on the command line, chmod changes the permissions of the pointed-to file. In contrast, chmod ignores symbolic links encountered during recursive directory traversals.

SETUID AND SETGID BITS

chmod clears the set-group-ID bit of a regular file if the file's group ID does not match the user's effective group ID or one of the user's supplementary group IDs, unless the user has appropriate privileges. Additional restrictions may cause the set-user-ID and set-group-ID bits of MODE or RFILE to be ignored. This behavior depends on the policy and functionality of the underlying chmod system call. When in doubt, check the underlying system behavior.

chmod preserves a directory's set-user-ID and set-group-ID bits unless you explicitly specify otherwise. You can set or clear the bits with symbolic modes like u+s and g-s, and you can set (but not clear) the bits with a numeric mode.

RESTRICTED DELETION FLAG OR STICKY BIT

The restricted deletion flag or sticky bit is a single bit, whose interpretation depends on the file type. For directories, it prevents unprivileged users from removing or renaming a file in the directory unless they own the file or the directory; this is called the restricted deletion flag for the directory, and is commonly found on world-writable directories like /tmp. For regular files on some older systems, the bit saves the program's text image on the swap device so it will load more quickly when run; this is called the sticky bit.

OPTIONS

Change the mode of each FILE to MODE. With --reference, change the mode of each FILE to that of RFILE.

-c, --changes
like verbose but report only when a change is made

-f, --silent, --quiet
suppress most error messages

-v, --verbose
output a diagnostic for every file processed

--no-preserve-root
do not treat '/' specially (the default)

--preserve-root
fail to operate recursively on '/'

--reference=RFILE
use RFILE's mode instead of MODE values

-R, --recursive
change files and directories recursively

--help display this help and exit

--version
output version information and exit

Each MODE is of the form '[ugoa]*([-+=]([rwxXst]*|[ugo]))+|[-+=][0-7]+'.

AUTHOR

Written by David MacKenzie and Jim Meyering.

REPORTING BUGS

GNU coreutils online help: <http://www.gnu.org/software/coreutils/>
Report chmod translation bugs to <http://translationproject.org/team/>

COPYRIGHT

or later <http://gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it. There is NO WARRANTY, to the extent permitted by law.

SEE ALSO
chmod(2)

Full documentation at: <http://www.gnu.org/software/coreutils/chmod>
or available locally via: info '(coreutils) chmod invocation'

GNU coreutils 8.28 January 2018 CHMOD(1)

Use chmod to make it possible to run the command ./semester rather than having to type sh semester. How does your shell know that the file is supposed to be interpreted using sh? See this page on the shebang line for more information.

```
side@ubuntu:/tmp/missing$ chmod u=rx semester
side@ubuntu:/tmp/missing$ ./semester
HTTP/2 200
content-type: text/html; charset=utf-8
server: GitHub.com
last-modified: Sat, 25 Apr 2020 11:12:41 GMT
etag: "5ea41b29-1abd"
access-control-allow-origin: *
expires: Sun, 03 May 2020 05:41:41 GMT
cache-control: max-age=600
x-proxy-cache: MISS
x-github-request-id: E6BE:7468:258856:2A45C3:5EAE573D
accept-ranges: bytes
date: Sun, 03 May 2020 06:35:32 GMT
via: 1.1 varnish
age: 110
x-served-by: cache-bom18227-BOM
x-cache: HIT
x-cache-hits: 1
x-timer: S1588487732.030397,VS0,VE1
vary: Accept-Encoding
x-fastly-request-id: 235b3b519d34164e8d17263cfda7ea0afd59c7db
content-length: 6845
```

Use | and > to write the "last modified" date output by semester into a file called last-modified.txt in your home directory.

side@ubuntu:/tmp/missing$ touch /home/side/last-modified.txt | date -r semester +%x > /home/side/last-modified.txt

side@ubuntu:/tmp/missing$ cat /home/side/last-modified.txt
05/02/2020

Write a command that reads out your laptop battery's power level or your desktop machine's CPU temperature from /sys. Note: if you're a macOS user, your OS doesn't have sysfs, so you can skip this exercise.

NOTE: Using a Virtual Machine so I just explored a little.

side@ubuntu:/sys/class/thermal/cooling_device0/device/thermal_cooling/stats$ cat time_in_state_ms
state0 1870053
state1 0
state2 0
state3 0
state4 0
state5 0
state6 0
state7 0