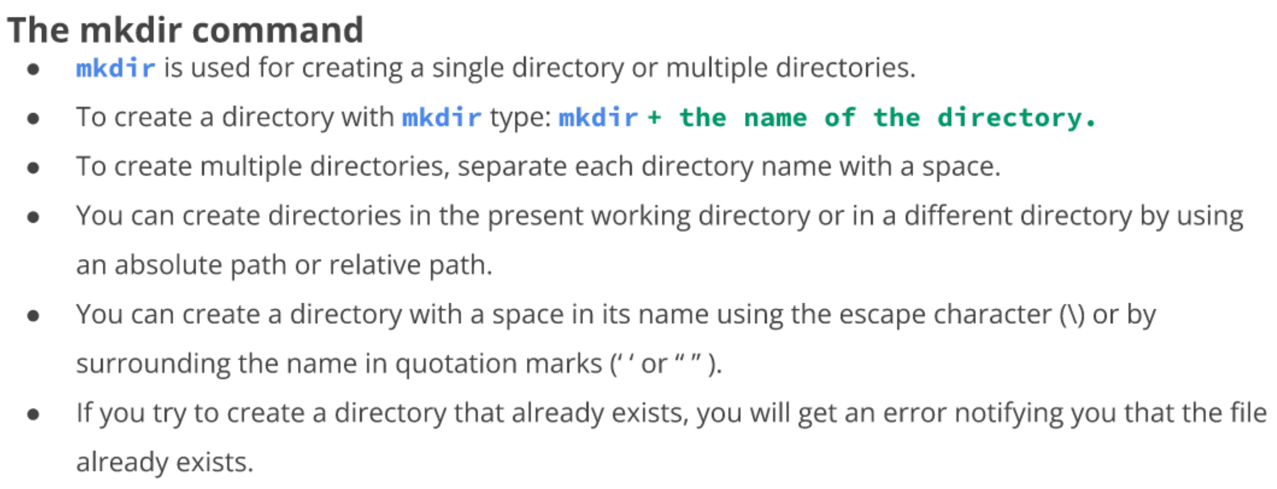
Week Report 5

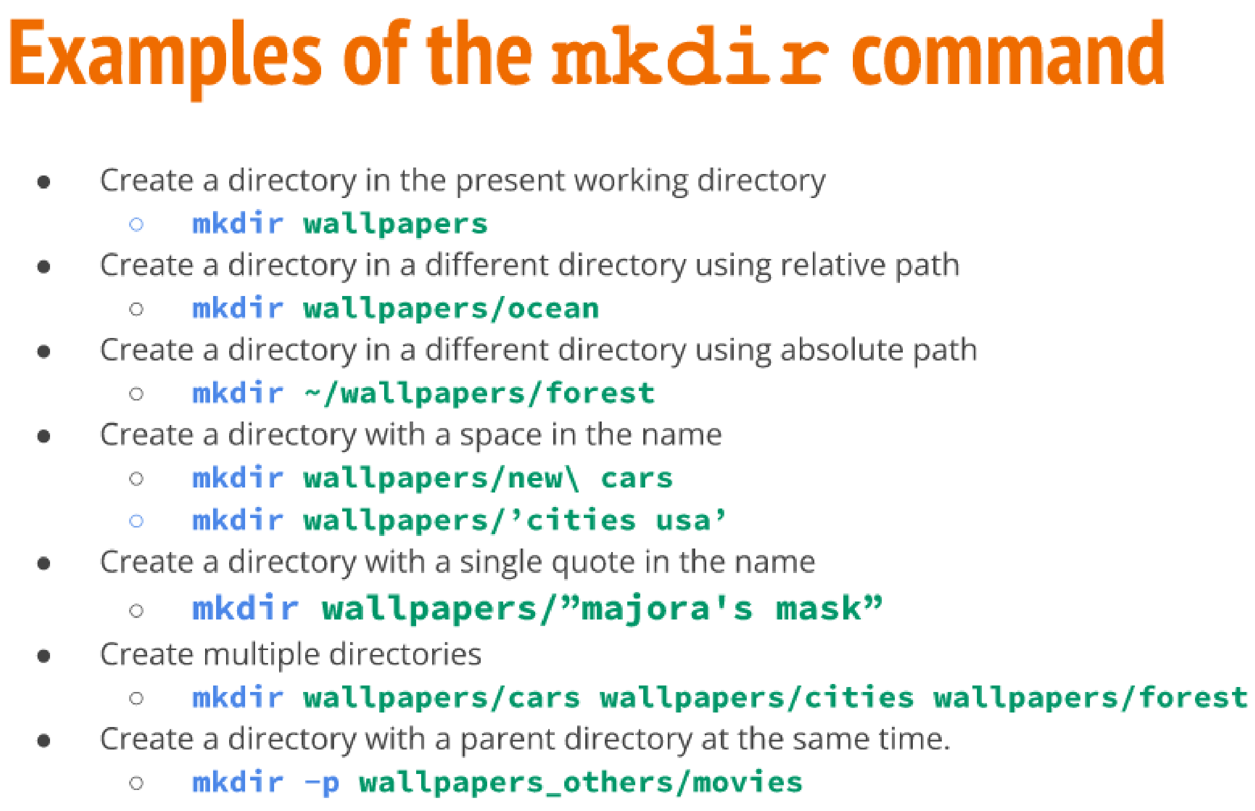
Definition, usage, and example of the following commands:

- **mkdir**:



The mkdir command

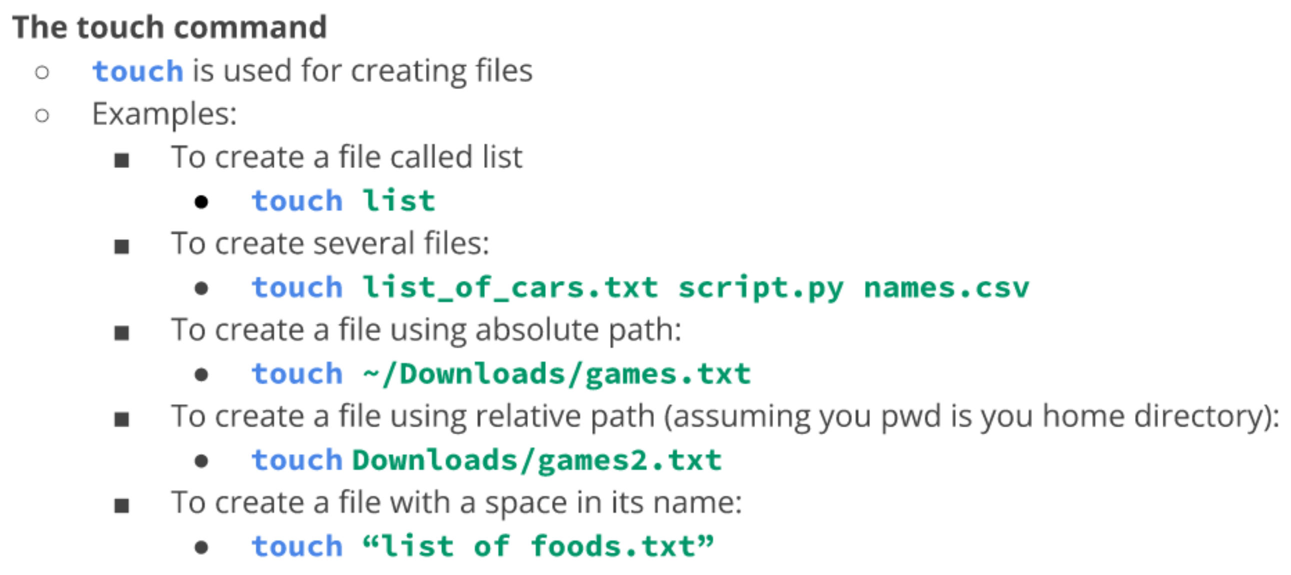
- `mkdir` is used for creating a single directory or multiple directories.
- To create a directory with `mkdir` type: `mkdir + the name of the directory.`
- To create multiple directories, separate each directory name with a space.
- You can create directories in the present working directory or in a different directory by using an absolute path or relative path.
- You can create a directory with a space in its name using the escape character (\) or by surrounding the name in quotation marks (‘ ‘ or “ ”).
- If you try to create a directory that already exists, you will get an error notifying you that the file already exists.



Examples of the `mkdir` command

- Create a directory in the present working directory
  - `mkdir wallpapers`
- Create a directory in a different directory using relative path
  - `mkdir wallpapers/ocean`
- Create a directory in a different directory using absolute path
  - `mkdir ~/wallpapers/forest`
- Create a directory with a space in the name
  - `mkdir wallpapers/new\ cars`
  - `mkdir wallpapers/’cities usa’`
- Create a directory with a single quote in the name
  - `mkdir wallpapers/”majora's mask”`
- Create multiple directories
  - `mkdir wallpapers/cars wallpapers/cities wallpapers/forest`
- Create a directory with a parent directory at the same time.
  - `mkdir -p wallpapers_others/movies`

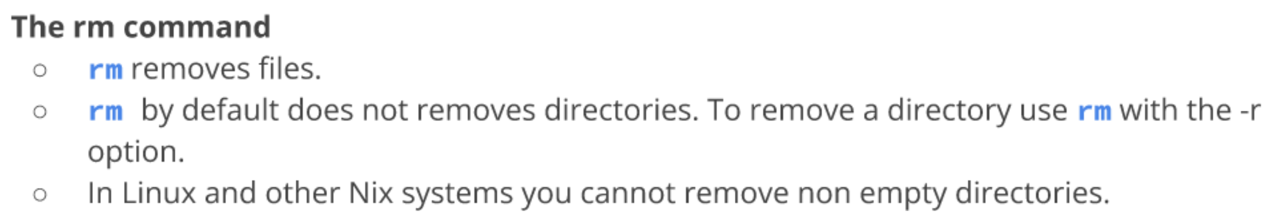
- **touch**:



The touch command

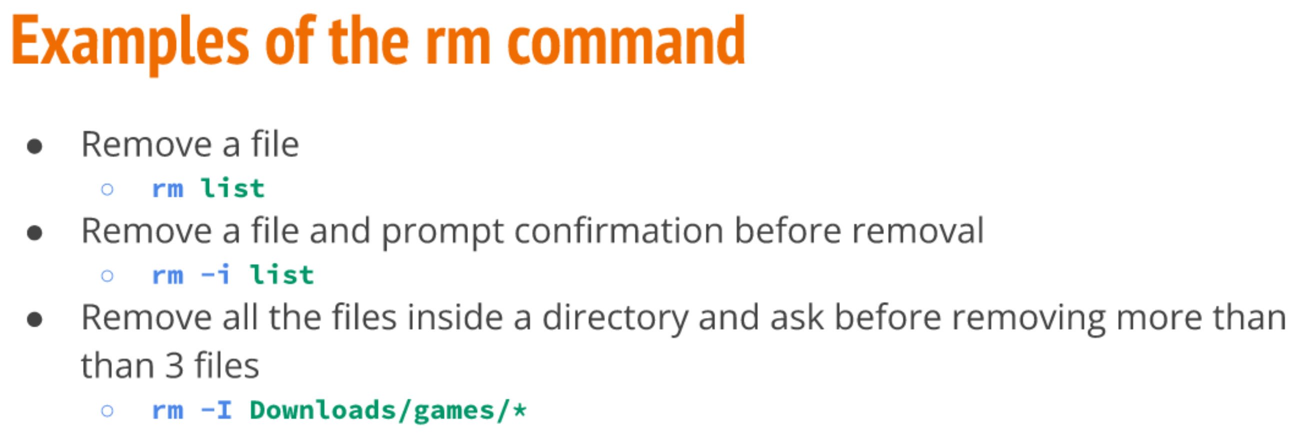
- `touch` is used for creating files
- Examples:
  - To create a file called list
    - `touch list`
  - To create several files:
    - `touch list_of_cars.txt script.py names.csv`
  - To create a file using absolute path:
    - `touch ~/Downloads/games.txt`
  - To create a file using relative path (assuming you pwd is you home directory):
    - `touch Downloads/games2.txt`
  - To create a file with a space in its name:
    - `touch “list of foods.txt”`

- **rm**:



The rm command

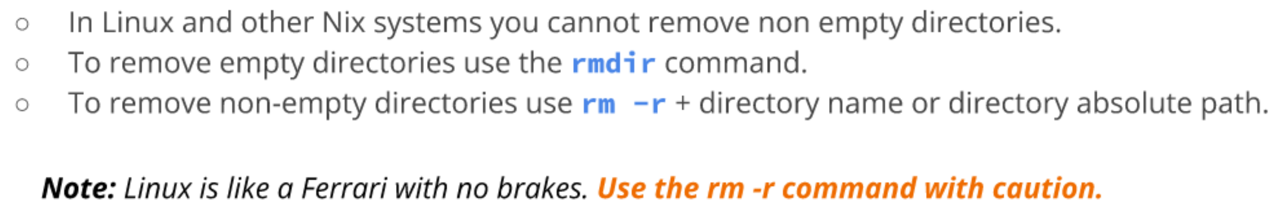
- `rm` removes files.
- `rm` by default does not removes directories. To remove a directory use `rm` with the -r option.
- In Linux and other Nix systems you cannot remove non empty directories.



Examples of the rm command

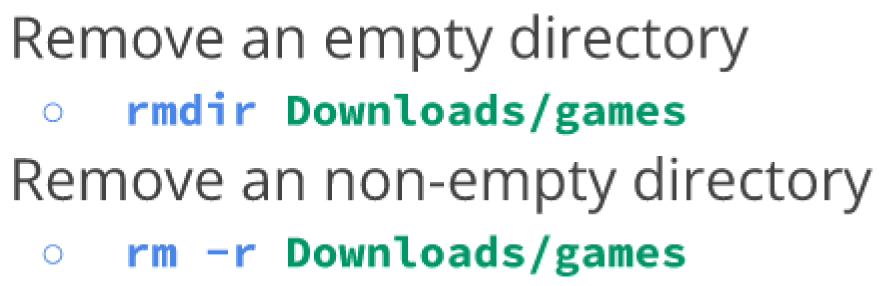
- Remove a file
  - `rm list`
- Remove a file and prompt confirmation before removal
  - `rm -i list`
- Remove all the files inside a directory and ask before removing more than than 3 files
  - `rm -I Downloads/games/*`

- **rmdir**:



- In Linux and other Nix systems you cannot remove non empty directories.
- To remove empty directories use the `rmdir` command.
- To remove non-empty directories use `rm -r` + directory name or directory absolute path.

Note: *Linux is like a Ferrari with no brakes.* ***Use the rm -r command with caution.***



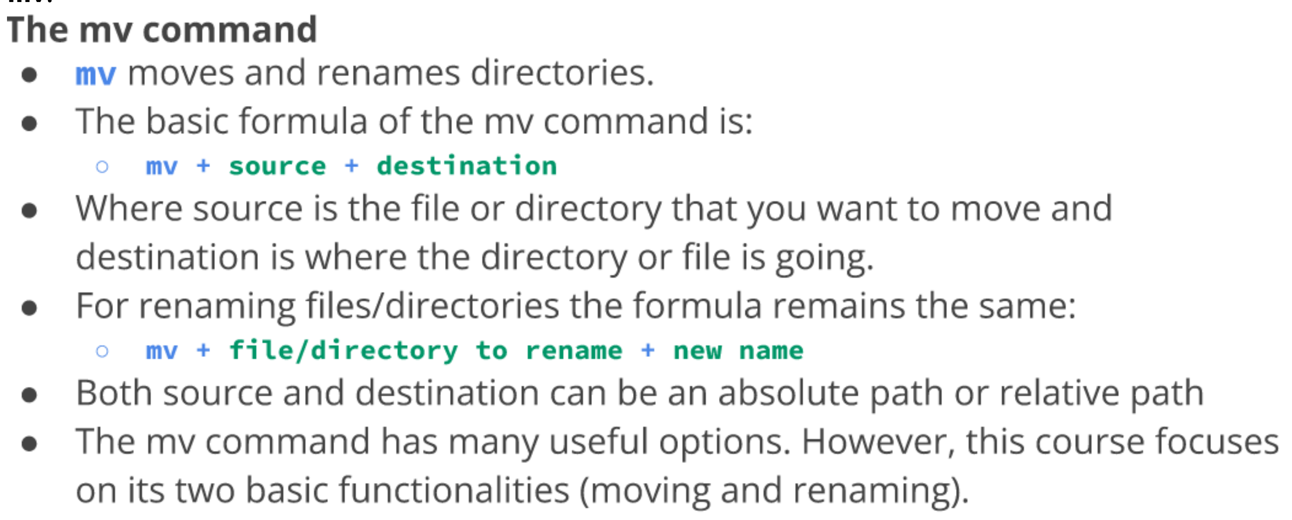
Remove an empty directory

- `rmdir Downloads/games`

Remove an non-empty directory

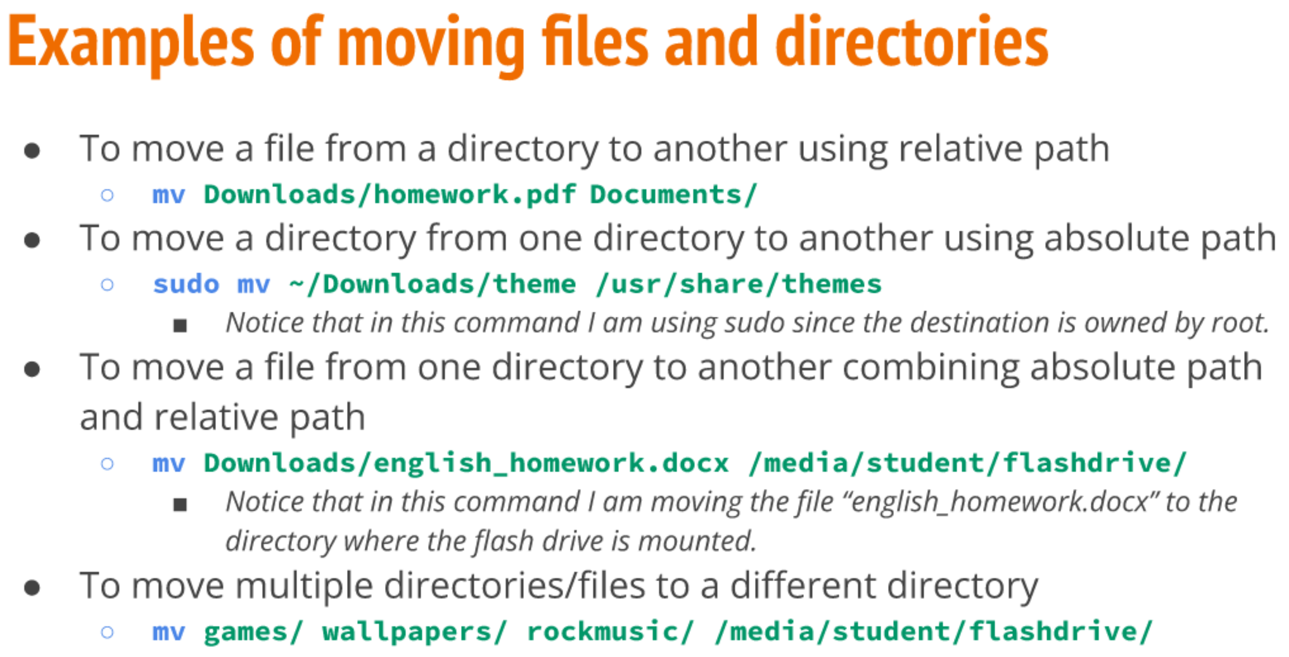
- `rm -r Downloads/games`

- **mv**:



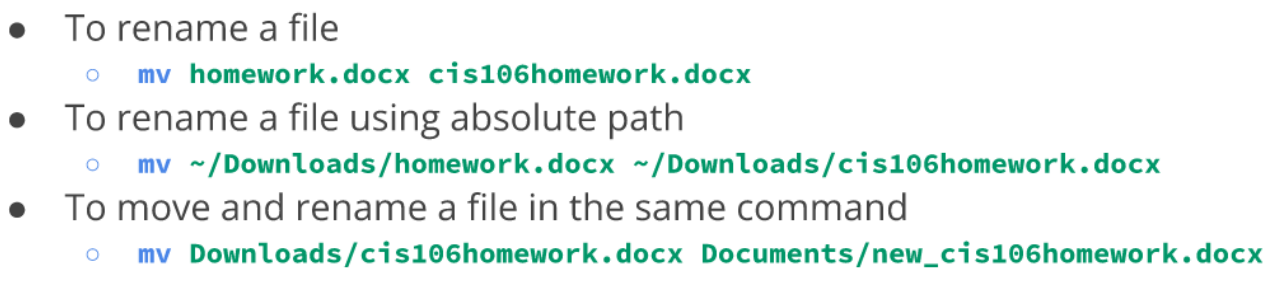
The mv command

- `mv` moves and renames directories.
- The basic formula of the mv command is:
  - `mv + source + destination`
- Where source is the file or directory that you want to move and destination is where the directory or file is going.
- For renaming files/directories the formula remains the same:
  - `mv + file/directory to rename + new name`
- Both source and destination can be an absolute path or relative path
- The mv command has many useful options. However, this course focuses on its two basic functionalities (moving and renaming).



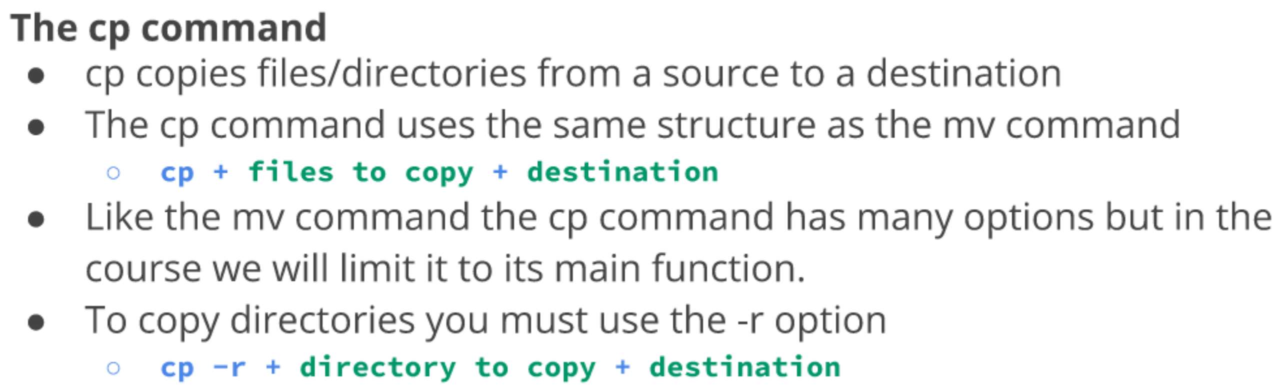
Examples of moving files and directories

- To move a file from a directory to another using relative path
  - `mv Downloads/homework.pdf Documents/`
- To move a directory from one directory to another using absolute path
  - `sudo mv ~/Downloads/theme /usr/share/themes`
    - *Notice that in this command I am using sudo since the destination is owned by root.*
- To move a file from one directory to another combining absolute path and relative path
  - `mv Downloads/english_homework.docx /media/student/flashdrive/`
    - *Notice that in this command I am moving the file "english_homework.docx" to the directory where the flash drive is mounted.*
- To move multiple directories/files to a different directory
  - `mv games/ wallpapers/ rockmusic/ /media/student/flashdrive/`



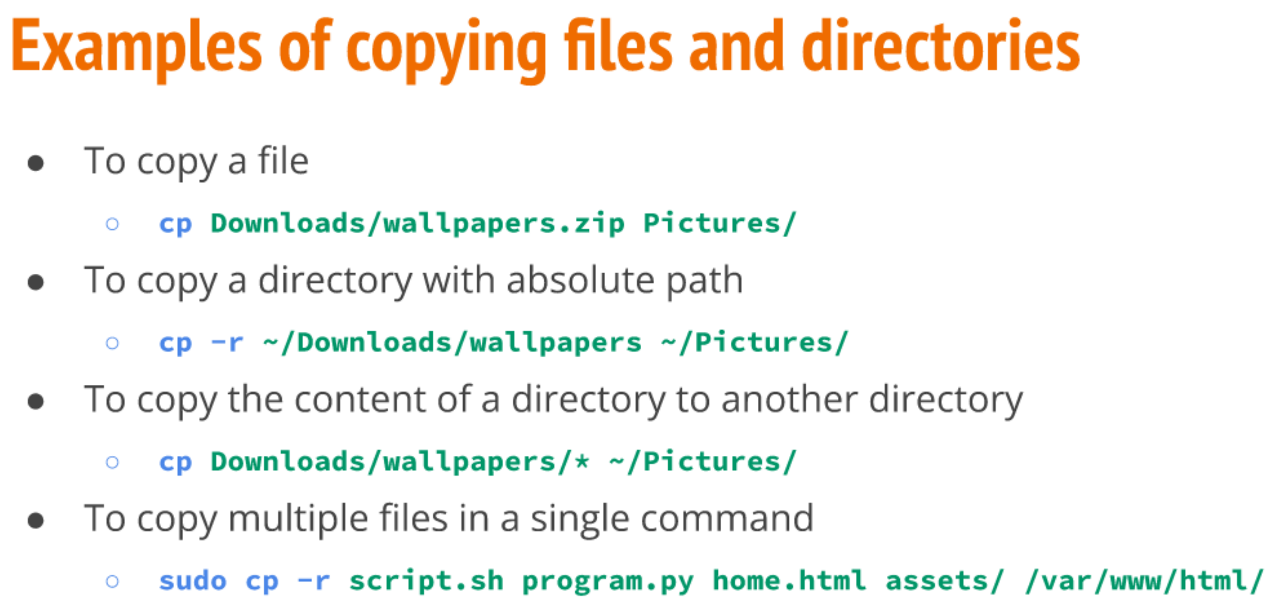
- To rename a file
  - `mv homework.docx cis106homework.docx`
- To rename a file using absolute path
  - `mv ~/Downloads/homework.docx ~/Downloads/cis106homework.docx`
- To move and rename a file in the same command
  - `mv Downloads/cis106homework.docx Documents/new_cis106homework.docx`

- **cp**:



The cp command

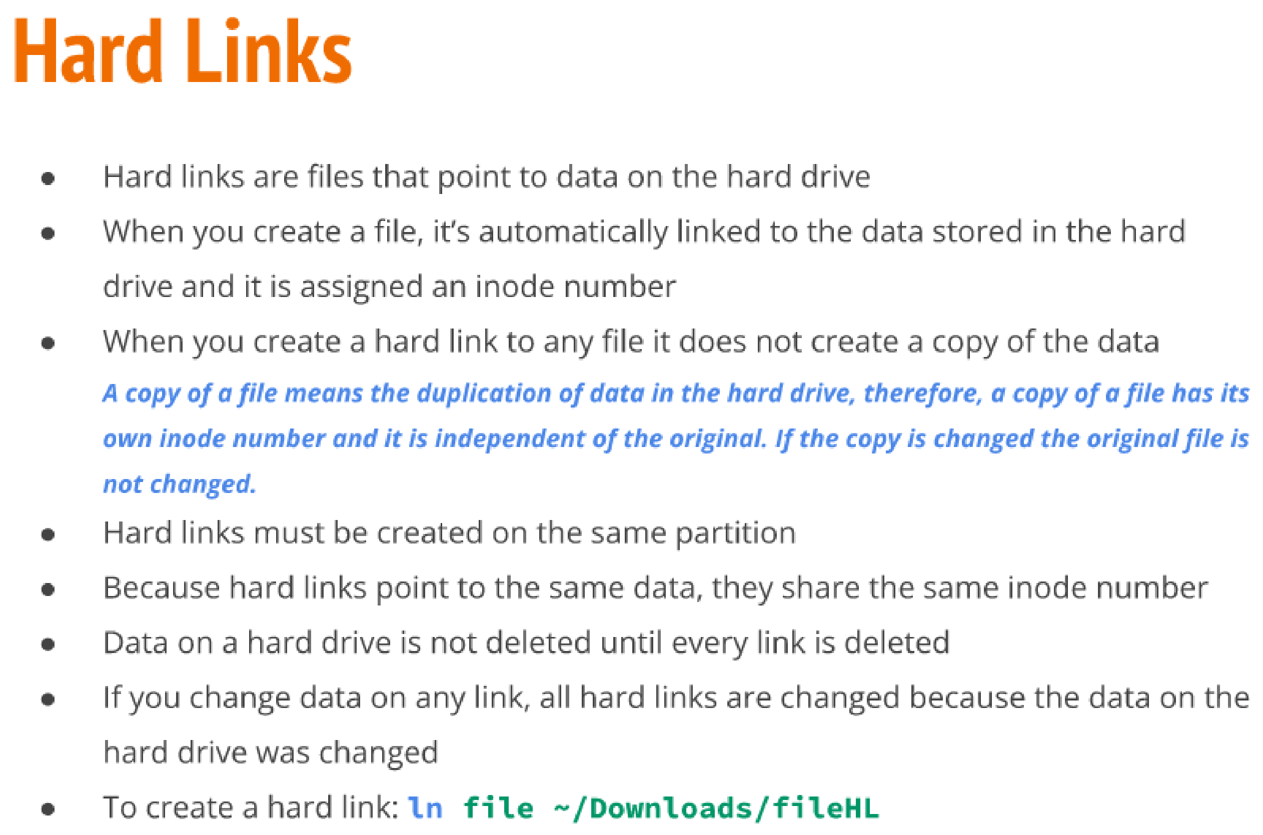
- cp copies files/directories from a source to a destination
- The cp command uses the same structure as the mv command
  - `cp + files to copy + destination`
- Like the mv command the cp command has many options but in the course we will limit it to its main function.
- To copy directories you must use the -r option
  - `cp -r + directory to copy + destination`



Examples of copying files and directories

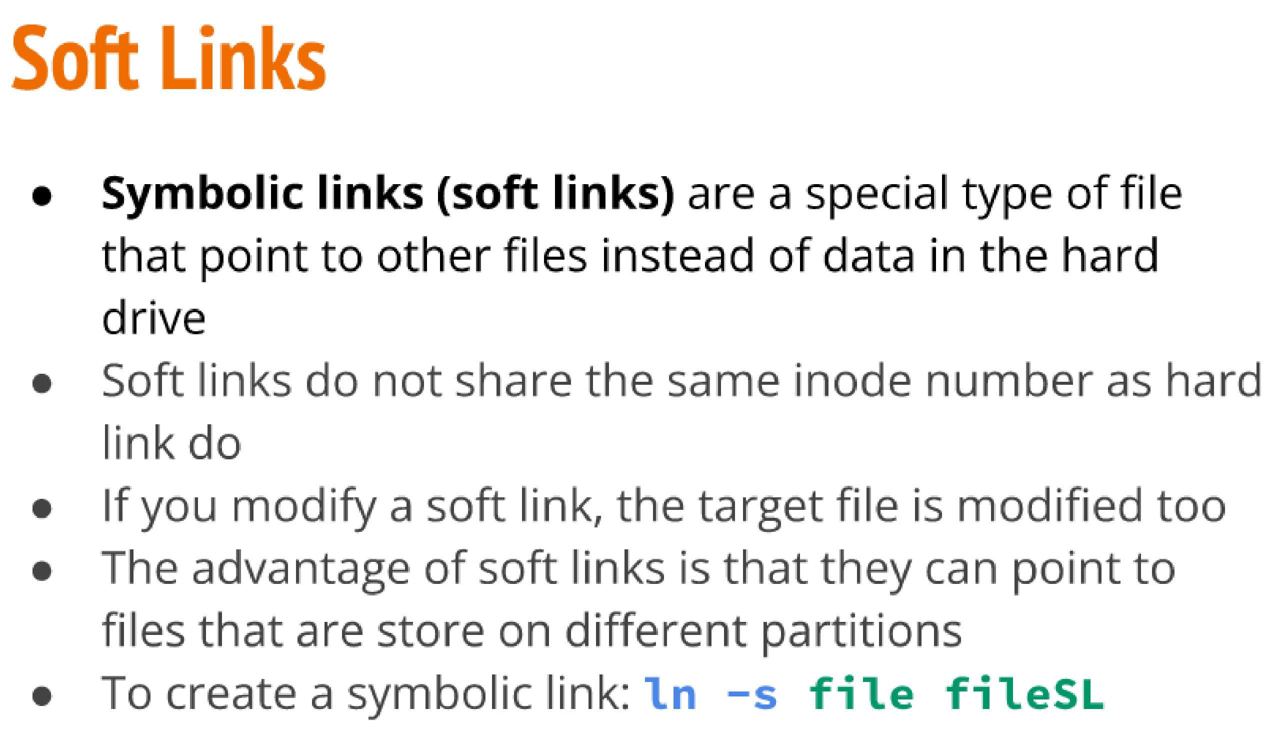
- To copy a file
  - `cp Downloads/wallpapers.zip Pictures/`
- To copy a directory with absolute path
  - `cp -r ~/Downloads/wallpapers ~/Pictures/`
- To copy the content of a directory to another directory
  - `cp Downloads/wallpapers/* ~/Pictures/`
- To copy multiple files in a single command
  - `sudo cp -r script.sh program.py home.html assets/ /var/www/html/`

- **ln**:



Hard Links

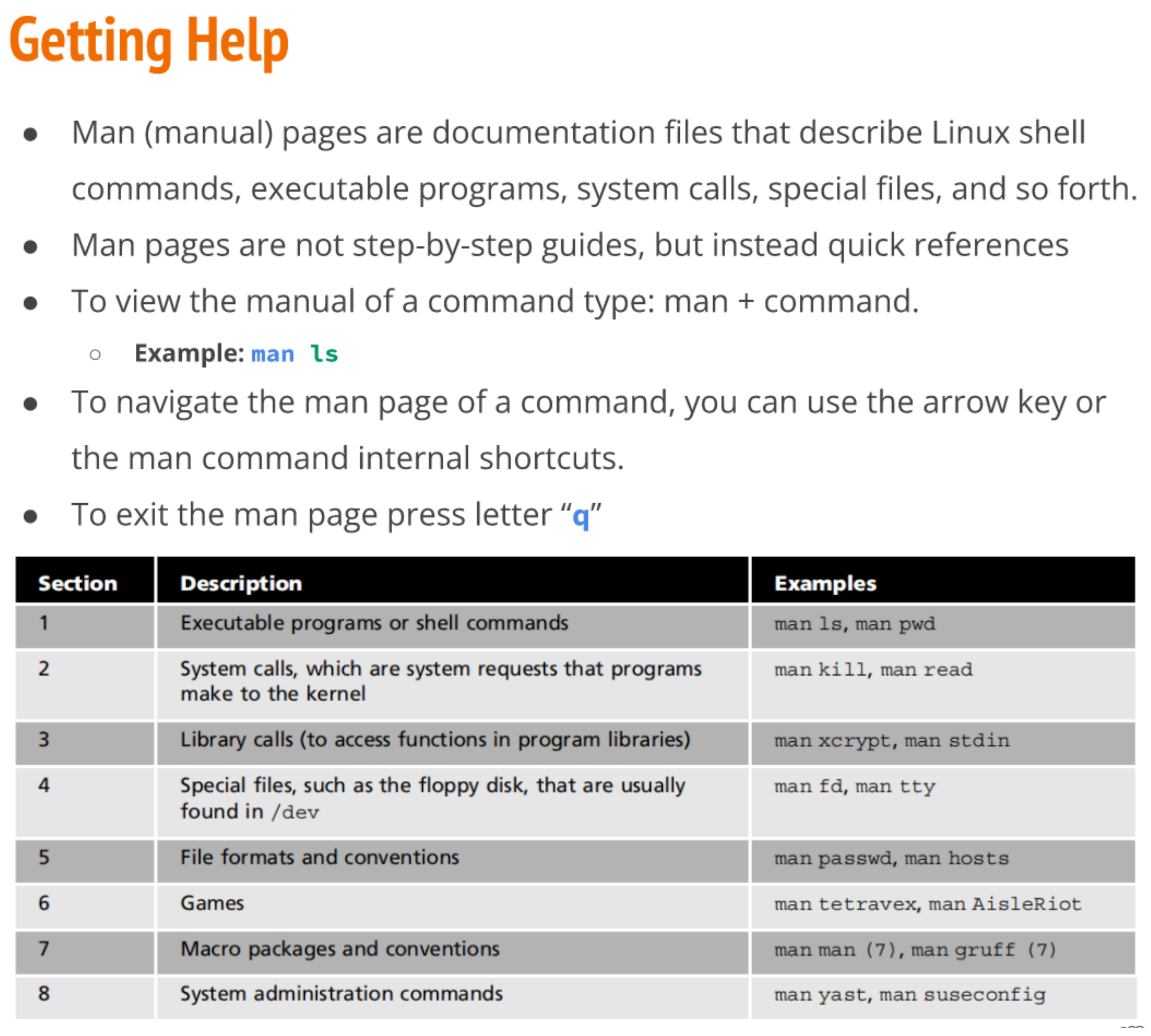
- Hard links are files that point to data on the hard drive
- When you create a file, it's automatically linked to the data stored in the hard drive and it is assigned an inode number
- When you create a hard link to any file it does not create a copy of the data
  A copy of a file means the duplication of data in the hard drive, therefore, a copy of a file has its own inode number and it is independent of the original. If the copy is changed the original file is not changed.
- Hard links must be created on the same partition
- Because hard links point to the same data, they share the same inode number
- Data on a hard drive is not deleted until every link is deleted
- If you change data on any link, all hard links are changed because the data on the hard drive was changed
- To create a hard link: `ln file ~/Downloads/fileHL`



Soft Links

- **Symbolic links (soft links)** are a special type of file that point to other files instead of data in the hard drive
- Soft links do not share the same inode number as hard link do
- If you modify a soft link, the target file is modified too
- The advantage of soft links is that they can point to files that are store on different partitions
- To create a symbolic link: `ln -s file fileSL`

- **man**:



Getting Help

- Man (manual) pages are documentation files that describe Linux shell commands, executable programs, system calls, special files, and so forth.
- Man pages are not step-by-step guides, but instead quick references
- To view the manual of a command type: man + command.
  - **Example:** `man ls`
- To navigate the man page of a command, you can use the arrow key or the man command internal shortcuts.
- To exit the man page press letter "**q**"

Section	Description	Examples
1	Executable programs or shell commands	`man ls`, `man pwd`
2	System calls, which are system requests that programs make to the kernel	`man kill`, `man read`
3	Library calls (to access functions in program libraries)	`man xcrypt`, `man stdin`
4	Special files, such as the floppy disk, that are usually found in `/dev`	`man fd`, `man tty`
5	File formats and conventions	`man passwd`, `man hosts`
6	Games	`man tetravex`, `man AisleRiot`
7	Macro packages and conventions	`man man (7)`, `man gruff (7)`
8	System administration commands	`man yast`, `man suseconfig`

Brace expansion and how to use it

Using Brace Expansion

- Brace expansion {} is not a wildcard but another feature of bash that allows you to generate arbitrary strings to use with commands.
- For example,
  - To create a whole directory structure in a single command:
    - `mkdir -p music/{jazz,rock}/{mp3files,vidoes,oggfiles}/new{1..3}`
  - To create a N number of files use:
    - `touch website{1..5}.html`
    - `touch file{A..Z}.txt`
    - `touch file{001..10}.py`
    - `touch file{{a..z},{0..10}}.js`
  - Remove multiple files in a single directory
    - `rm -r {dir1,dir2,dir3,file.txt,file.py}`