Basic file operations in Unix

Goals for this part of class

Learn how to create, delete, rename, and move both files and directories

Learn how to update files without opening them

Learn how to view the contents of files

Become familiar working with: `mkdir`, `rmdir`, `touch`, `rm`, `cp`, `mv`, `echo`, and `cat`

With great power…

Unix assumes you know what you are doing

- It gives you enormous power and flexibility, but provides minimal supervision

Most file operations are executed **without safety checks**

- However, you can use flags to add checks to some file operations

There is **no undo** for file operations, **no recycle/trash folder** for deleted files

- Always work with duplicates of original data and back up of your data and code

There is typically **no feedback** unless an error occurs

- Until you are very comfortable, always check results (e.g., with `ls` and `pwd`)

Creating directories

Use `mkdir` to create new directories (name is a contraction of **m**a**k**e **dir**ectory)

`mkdir practice`	creates a new directory
`mkdir ~/Data/practice`	creates a new directory at specified location
`mkdir practice temp`	creates two new directories
`mkdir -p thesis/chapter_01`	creates new and nested directories

Notes:

The shell will return an error message if the directory already exists

By default, file operations refer to the current directory

Multiple directories can be created at once, but they will not be nested

To create nested directories in a single step, the `-p` option is required

Deleting directories

Use `rmdir` to delete an existing empty directory

`rmdir practice`	deletes a directory
`rmdir data/week_02 data`	deletes two directories in the order given

Notes:

The shell will return an error if the directory does not exist or if it contains any files

Delete nested directories *before* deleting parent directories (second example)

You can also use `rm` to delete directories, but this is not recommend

Creating empty files

Use `touch` to create an empty file or to update the timestamp on an existing file

- If the specified file(s) does not exist, creates an empty file
- If the specified file(s) exists, simply updates the timestamp of most recent access

`touch chapter_1.txt`	creates an empty file
`touch chapter_{1..100}.txt`	creates 100 empty files (`chapter_1.txt`, etc.)

Notes:

- You can create multiple files at once and can create files in other locations
- Unix keeps separate timestamps for each file's creation and most recent access
- The second command is an example of an **expansion** (more on these later)

Deleting files

Use `rm` to delete files (name is a contraction of **re**mo**v**e)

```
rm temp_1.txt
```

removes a single file

Notes:

The shell will return an error message if the file does not exist

Multiple files can be specified (separate with spaces)

Files at other locations can be specified (provide a path)

Copying files

Use `cp` to make copies of files (name is a contraction of **c**o**p**y)

`cp temp_01.txt temp_backup.txt`	makes a copy in the same directory
`cp temp_01.txt /data`	makes a copy in `/data`, using same name
`cp temp_01.txt /data/test.txt`	makes a copy in `/data`, using different name
`cp temp_01.txt temp_02.txt /data`	copies 2 files to `/data`, using same names

Notes:

The first argument is the original file, the second argument is the new file

The new file will be placed in the current directory unless a path is provided

The new file name must be different if copying into the current directory

Multiple files can be copied to a single different location (last example)

Also works with directories (use the `-R` option to copy any enclosed directories)

Moving and renaming files

Use `mv` to move and/or rename files (name is a contraction of **mov**e)

`mv temp_01.txt /data`	moves file to different location
`mv temp_01.txt perm_01.txt`	renames file without moving it
`mv temp_01.txt /data/perm_01.txt`	moves file to different location and renames
`mv data data_original`	renames directory

Notes:

The first argument is the file, the second argument is the new location or name

The file can be renamed while moving (second example)

Also works with directories

Tips for copying, moving, and renaming files

Be aware:

The shell will overwrite existing files if they have the same name!

The shell will give you no warning that it did this!!

`rm` has options that provide a layer of security:

`rmdir -i temp_01.txt`	prompts y/n before deleting
`rmdir -v temp_??.txt`	reports on each file successfully deleted

`mv` and `mv` have options that provide a layer of security:

`mv -i temp_01.txt /data`	prompts y/n before overwriting
`mv -n temp_01.txt data_01.txt`	prevents overwriting

Writing output to a file

Use the > operator to **redirect** output to a file instead of the terminal

```
touch temp_1.txt
ls -l > temp_1.txt
```

creates an empty file

writes the directory listing to the file

Use the >> operator to redirect output to a file and append (rather than over-writing):

```
pwd >> temp_1.txt
date >> temp_1.txt
```

appends the current directory name

appends a timestamp

Writing strings to the terminal or to a file

Use the `echo` command to write strings

```
echo hello world
echo hello world >> temp_1.txt
echo “hello world” >> temp_1.txt
```

outputs `hello world` at the terminal

appends `hello world` to the file

appends `hello world` to the file

Note:

Quotes are not needed unless using escape characters or inserting variables

`echo` can be surprisingly useful as you get familiar with Unix

Viewing the contents of text files

Use the `cat` command to view files (name is short for con**cat**enate)

`cat temp_1.txt`	outputs contents of `temp_1.txt` to the terminal
`cat temp_1.txt temp_2.txt`	outputs contents of both files, in order
`cat -n temp_1.txt`	adds line numbers to output

Notes:

- There are other ways to view files (e.g., `head`, `tail`, `less`)
- `cat` is very useful in pipes for passing the contents of a file to the next command

Exercise

Create a folder called `thesis_project_1`

Create a subfolder within `thesis_project_1` called `original data`

Navigate to the new subfolder

Create 10 text files called called `data_01.txt` through `data_10.txt`

Delete the file `data_05.txt`

Without changing location, insert a listing of your home directory into `data_01.txt`

View the contents of `data_01.txt` with line numbers

Tip: at each step, use `ls`, `pwd`, and/or `cat` to check your results!