15. Grep

commands table

command	application

01. what actually matters in this section

- `important`: grep basics, common grep options
- `useful`: extended regex syntax
- `nice to have`: the crazy looking URL regular expansion at the end of the regular section

02. Introducing Grep Command

- `grep` command searches for patterns in each file's contents. It will print each line that matches a pattern we provide
- examples:

1. `grep "chicken" animals.txt` prints each line from the `animals.txt` file that contains the pattern `"chicken"`

- The command by default is case sensitive
- `-i` can be used as an option to perform a case-insensitive search

word search

- `-w` option is used to ensure that grep only matches words, rather than fragments located inside of other words
- example:
  1. `grep -w "cat" book.txt` would match `cat` but not `catheter`

03. Grep Recursive Search

- `-r` option is used to perform a recursive search which will include all files under a directory subdirectories and their files, and so on.
- If a directory is not specified then, grep will search the current working directory.
- example:
  1. `grep -r "chicken"` will search the current working directory and any nested directories for lines that contain `"chicken"`
  2. `grep -ri "parm[ae]san" Mealdiary/` searches for files containing both `parmasan` and `parmesan`

04. Grep options

- `-c` to print the number of matches
- `-A<number>` prints number of lines after match
- `-B<number>` prints the number of lines before the match

- `-C<number>` prints the number of lines before and after the match together
- `-n` prints line number for the output
- `-m<number>` print first `n` matches based on number passed.

05. Grep & regular expressions

- Regular expressions help to match complex patterns. It can also be used with grep.
-

regex syntax	application
`.`	matches any single character
`^`	matches the start of a line
`$`	matches the end of a line
`[abc]`	matches any character in the set
`[^abc]`	matches any character NOT in the set
`[A-Z]`	matches any character in the range
`*`	repeat the previous expression 0 or more times
`\`	escape meta characters

06. Grep Extended

Extended regex syntax	application
`?`	match 0 or 1 of the preceding expression
`{<number>}`	num of characters preceding to be present from the passed input range. e.g: `grep [aeiou]{3} -M <filename.txt>` will print all the cases where there is a combination of 3 characters including `a,e,i,o,u`

- we use `-E` option with regular `grep` command to use extended grep.
- It provides more flexible ways of regular expressions.
- example:
    1. the print word that matches to `bird` and `birds` where `s` is an optional character whereas `bird` is necessary.

```
grep "birds?" -wE <filename.txt>
```

07. Piping to grep

- we can also use pipe and grep together to use the regex as well so that we can do more efficient searching
- example:
    1. `ps -aux | grep hermione` : `ps -aux` gives the list of processes being executed in the system but when it is piped with the `grep hemoine`, it will look for the results containing `hermoine` which will give only the list of processes having it.