20.03.2024

wc → word count

- grep → pattern matching

  -E → enabling extended regular expressions
  "less backslashes"

  -i → ignore case sensitivity.

ex cat /etc/passwd | grep

  -v → invert match

  -w → matches only whole word

  ↳ P.S.: do not use ^, $ in the regex :)

  -c → count of matching lines of each input

  -o → shows only matches (each one on a different line)

  -q → quiet; does not write to the console.

  -n → prefix each line in the output with the line at which the match is in the file.

  - A number
  - B number

  } shows the number of lines after / before the match.

REGEX

^ -> beginning of line

$ -> end of line

. -> wildcard that replaces any char.

repeators

* -> 0 or more times

? -> 0/1 times

+ -> 1 or more times

used after an expression and affects the expression <u>before</u> it

{n, m} -> must appear at least n times and at most m times

{ ,m} -> must appear at most m times

{n, } -> must appear at least n times

\ -> cancels the special ability of a character

[ ] -> used to search

[0-9] -> digits

[a-z] -> small letters

[A-Z] -> capital letters

[0-9a-z] ->

[aeiou]

[AEiOU]

[AaEeIiOoUu]

[.,:;] obs: no need of "\" when we use special chars.

[^0-9] -> not digit

reverse matching -> negates EVERYTHING inside the [...]

| -> logical or ex ([0-9]|$)

- sed -e "..." input file

-e -> expression

flags (optionals)

"s/regexp/replacement/gi"

s -> substitute (will not substitute in the file, it will substitute in the output only!)

g -> global replacement (not only first occurrence)

i -> ignore case sensitivity

"s/(^[0-9]+)(.*[^0-9])([0-9]+$)/\3\2\1/g

- awk

Records -> default separator '\n'

Field -> default separator ' '

awk '<condition> {code}'

optional

-> for each of the lines for which the condition is satisfied

-f take the code from a file

Obs: awk can take multiple files as input -f... -f....