Parancssor, reguláris kifejezések, gyakorisági lista

(2022. 02. 28.)

Mittelholcz Iván

---

1. Átirányítás

Standard csatornák:

- STDIN: standard bemenet, billentyűzet
- STDOUT: standard kimenet, terminál
- STDERR: standard hiba kimenet, terminál

Átirányítás és pipeline:

- `command <filename`: fájl → stdin
- `command >filename`: stdout → fájl (felülír)
- `command >>filename`: stdout → fájl (appendál)
- `command 2>filename`: stderr → fájl
- `command1 | command2`: 1. parancs STDOUT → 2. parancs STDIN

2. Parancsok

Írás a STDOUT-ra

- `echo 'string'`: *text* printelése a STDOUT-ra
- `cat`: STDIN másolása STDOUT-ra
  - `cat file1 file2 ...`: fájlok konkatenálása (és másolása a STDOUT-ra)

Keresés

`grep <kifejezés>`: Fájl vagy a stdin szűrése, a kifejezést tartalmazó sorokra.

- `-E`: *basic* helyett *extended* regexek használata
- `-i`: ignore case
- `-v`: fordított működés - a nem illeszkedő sorokat átengedi, az illeszkedőket kiszűri
- `-r <könyvtár>`: rekurzívan keres a könyvtár minden alkönyvtárjában
- `-f <fájl>`: nem kell kifejezést megadni, helyette a megadott fájlból olvassa ki a keresett kifejezéseket (egy sor = egy kifejezés)

Csere

`sed "s/mit/mire/g"`: Kifejezés keresése és cseréje fájlban vagy STDIN-en.

- `-E`: *basic* helyett *extended* regexek használata

Rendezés

`sort`: Sorok ábécé szerinti rendezése.

- `-n`: numerikus rendezés (a default lexikális helyett, pl. $10 > 9$)
- `-r`: fordított sorrend

Egyelés

`uniq`: Dupla sorok szűrése. Csak az egymást követő azonos sorokat egyeli, ezért előtte szükséges lehet rendezni.

- `-c`: a sorok elé írja, hány darab volt belőlük

További hasznos parancsok

- `wc`: (*word count*) sorok (`-l`), szavak (`-w`) és karakterek (`-c`) számlálása
- `cut`: Oszlopok szelektálása. Default: `TAB`-ok mentén.
  - `-d"<char>"`: határoló karakter (pl. `-d";"`, `-d" "`)
  - `-f<nums>`: oszlopszám (pl. `-f2`, `-f2,4`, `-f2-4`)

3. Reguláris kifejezések

Több szabványos "nyelvjárás" létezik (de ezeken túl az egyes programok is működhetnek különbözőképpen):

- *basic* (BRE): kerek és kapcsos zárójelek eszképelve, nincs `+`, `?` és `|`
- *extended* (ERE): ezt fogjuk tanulni
- *perl-kompatibilis* (PCRE): lustaság és sok minden más (Python-ban is ilyesmi van)

A *grep* és a *sed* alapból a BRE-t használják, a *sed -r* vagy *sed -E* illetve a *grep -E* vagy *egrep* viszont már a kiterjesztett regularis kifejezéseket használja.

Szintaxis

Pozícióra (nulla karakterre) illesztés:

- `^`: string / sor elejére illeszkedik
- `$`: string / sor végére illeszkedik

Egy karakterre illesztés:

- `.`: bármilyen karakterre illeszkedik
- különleges karakterre azt iszképelve lehet illeszteni, pl. `\.` illeszkedik a .-ra.
- `x`: literális karakter, saját magára illeszkedik
- `[ ]`: a zárójelen belül felsorolt karakterek valamelyikére illeszkedik, pl. `[ab]` illeszkedik az `a` vagy a `b` karakterre, másra nem.
  - Megadható tartomány is, pl `[a-z]` illeszkedik az ASCII kisbetűkre, `[0-9]` pedig a számjegyekre.

  - Ha a kötőjelet is be akarjuk venni a felsorolt karakterek közé, akkor a felsorolás elejére vagy végére kell írni.
  - A szögletes zárójelen belül más karakterek elveszítik speciális jelentésüket, pl. `[.]` egy literális pontra illeszkedik, nem pedig bármire.
- `[^ ]` illeszkedik a zárójelen belül fel nem sorolt karakterek valamelyikére. Megadható tartomány is, pl. `[^A-Z0-9]` illeszkedik minden karakterre, ami nem ASCII nagybetű és nem is számjegy.

Változó hosszúságú illesztések (mindig mohó):

- `|`: Alternáció, az előtte vagy az utána következő regex valamelyikére illeszkedik, pl. `abcd|xyz` illeszkedik *abcd*-re és *xyz*-re is. Alternációt lehatárolni zárójellel lehet, pl. `ab(cd|xy)z` illeszkedik az `abcdz` és az `abxyz` stringekre.
- `()`: A zárójelen belüli kifejezés megnevezett csoport lesz, amire később hivatkozni lehet. Általában egymásba ágyazhatók, de nem fedhetnek át. A `(a.(.a))` illeszkedik pl. az *abba* stringre. Zárójelre hivatkozni backslash-sel lehet: `\(` és `\)`
- `\n`: Hivatkozás egy csoportra. Pl. `(a.(.a)) \2 \1` illeszkedik az *abba ba abba* stringre.
- `?`: nulla vagy egy az előző karakterből / csoportból
- `*`: nulla vagy bármennyi az előző karakterből / csoportból
- `+`: legalább egy az előző karakterből / csoportból
- `{m,n}`: minimum m, maximum n darab az előző karakterből / csoportból.
  - `{m,}` alakban csak a minimumot is megadhatjuk (a maximum ekkor bármennyi lehet, hasonlóan a `*`-hoz).
  - `{,n}` alakban csak a maximumot is megadhatjuk (a minimum ekkor nulla, hasonlóan a `*`-hoz)
  - `{m}`

Műveletek precedenciája: *csillag* > *konkatenáció* > *alternáció*

Különleges karakterek:

- `\n`: új sor (new line)
- `\t`: TAB
- `\s`: whitespace karakterek
- `\S`: nem whitespace karakterek
- `\w`: szóalkotó karakterek (számjegyek, betűk és alulvonás)
- `\W`: nem szóalkotó karakterek

Egy-egy reguláris nyelv általában sokféleképpen megadhatók regexekkel (pl. `a+` = `aa*`), nincs igazán jó egyszerűsítő módszer, ezért érdemes jól megírni a regexeket!

4. Gyakorlatok

- Szógyakorisági lista
  - Hozzunk létre egy fájlt, ami valami szöveget tartalmaz.
  - Alakítsuk át a szöveget “egy sor = egy szó” formátumra

  - a punktuációkat dobjuk ki, üres sorokat töröljük
  - A kapott kimenetet alakítsuk tovább szógyakorisági listává: egy sor egy szót és az ő gyakoriságát tartalmazza. Egy szó (type) csak egyszer forduljon elő. A lista gyakoriság szerint fordítottan legyen rendezve.
  - hozzunk létre egy *stopwords.txt* fájlt a nem számolandó szavaknak.
  - Szűrjük a gyakorisági listánkat *stopwords.txt*-vel (`grep -f` + egy trükk)
- Hány szó kezdődik *a*-val egy listában?
- Hány szó kezdődik *a*-val folyó szövegben?
- Hány három betűs szó van a szövegben?
- Hány zárójelpár van a szövegben?
- Töröljük egy szövegből a zárójelpárokat (tartalmukkal együtt). Vigyázzunk a mohósággal!
- Szorgalmi: Dobjuk ki a HTML / XML *tag*-eket egy fájlból. Egészítsük ki ezzel a lépéssel a gyakorisági lista előállító parancsunkat, hogy HTML fájlokra is működjön.

5. Infók

- Software Carpentry: The Unix Shell- kezdőknek is jó tutorial
- online shell: https://www.tutorialspoint.com/execute_bash_online.php
- Windows Subsystem for Linux: https://docs.microsoft.com/en-us/windows/wsl/install