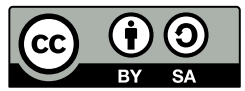
Filtrai, mikrokalbos ir shell-skriptai

Saulius Gražulis

2009 ruduo

Vilniaus universitetas, Matematikos ir informatikos fakultetas
Informatikos institutas

Šį skaidrių rinkinį galima kopijuoti, kaip nurodyta Creative Commons Attribution-ShareAlike 4.0 International licenzijoje



Filtrai

- Filtras – tai programa, kuri skaito savo duomenis iš savo standartinės įvesties ir rašo rezultatus į standartinę išvestį
- Komanda `cat` gali veikti kaip filtras:
  - `sh> echo "I Dalis" | cat - I-d.txt pabaiga.txt > pirma-dalis.txt`

Get Regular ExPression

grep

- Usage: grep [OPTION]... PATTERN [FILE]...
- komanda `grep` ieško reguliarią išraišką PATTERN atitinkančių eilučių savo standartinėje įvestyje arba savo failuose–argumentuose.
- Pvz.:
  - `sh> grep root /etc/passwd`
  - `root:x:0:0:root:/root:/bin/bash`
  - `sh> cat 1.dat 2.dat 3.dat | grep AVERAGE`
  - `...`
- `grep` reguliariose išraiškose sutinkama daug shell specialių simbolių, todėl rekomenduojama visą išraišką rašyti kabutėse.

`grep` reguliariosios išraiškos

- Paprasti simboliai (raidės, skaičiai) atitinka juos pačius
- Taškas („.“) – atitinka bet kokį vieną simbolį
- Laužtiniuose skliausteliuose („[]“) nurodyti simboliai arba jų ruožas – atitinka bet kurį nurodytai aibei priklausantį simbolį
- Žvaigždutė („*“) – modifikuoja prieš ją einančią reguliarią išraišką, ir leidžia ją atitikti 0 arba daugiau kartų

grep reguliariosios išraiškos

- Klaustukas („?“) – modifikuoja prieš ją esančią reguliarią išraišką, ir leidžia jai atitikti simbolį nė vieno (0) arba vieną (1) kartą
- „Rodyklytė į viršų“ („^“) – atitinka eilutės pradžią
- Dolerio ženklas („$“) – atitinka eilutės pabaigą
- Atbulas brūkšnelis („\“) – pažymi, kad po jo einantis simbolis praranda savo specialią reikšmę

grep naudojimo pavyzdžiai

- `sh> grep '1 dalis' knyga.txt`
- `sh> grep 'ATOM ' 1knv.pdb`
- `sh> grep '^ATOM' 1knv.pdb`
- `sh> grep '^ATOM ' 1knv.pdb`
- `sh> grep '^SCALE[123]' 1knv.pdb`
- `sh> grep '^SCALE[1-9]' 1knv.pdb`

PDB failų apdorojimo *x komandomis pavyzdžiai

PDB faile kiekviena duomenų eilutė (“įrašas”) pažymėta 6 simbolių ilgio raktiniu žodžiu eilutės pradžioje; duomenų reikšmės išdėstytos fiksuotose kolonėlėse (pvz. grandinės pavadinimas yra 22 kolonėlėje).

- Pasirinkti tik ATOM įrašus iš PDB failo:

```
sh> grep '^ATOM␣␣' 1knv.pdb
sh> grep "^ATOM␣␣" 1knv.pdb
```

- Pasirinkti tik CRYST, ATOM, HETATM ir END įrašus iš PDB failo (-E argumentas reiškia “use extended regular expression”):

```
sh> grep -E '^(CRYST1|ATOM␣␣|HETATM|END)' \ 1knv.pdb
```

- Sužinoti, kokios yra grandinės PDB faile:

```
sh> grep -E '^(ATOM␣␣|HETATM)' 1knv.pdb \
| cut -b 22-22 | sort | uniq
```

sed mikrokalba

- Usage: sed [OPTION]... script [input-file]...
- `sh> echo -e "vienas\ndu\ntrys" > tekstas.txt`
- `sh> cat tekstas.txt`

```
vienas
du
trys
```

- `sh> sed -e 's/vienas/1/' tekstas.txt`

```
1
du
trys
```

- `sh> sed -e 's/d./2/' tekstas.txt`

```
vienas
2
trys
```

`awk` programavimo kalba

Aho, Weinberger, Kernighan

Programa `awk` leidžia, kaip ir `grep`, atrinkti failo eilutes pagal atitikimą reguliarioms išraiškoms (užrašomoms tarp įstrižų brūkšnelių, `/.../`). Be to, kiekvienai atitikusiai eilutei galime įvykdyti nedidelę programėlę (užrašytą tarp figūrinių skliaustelių, `{...}`).

- Usage: awk [POSIX or GNU style options] -f progfile [–] file ...
  Usage: awk [POSIX or GNU style options] [–] 'program' file ...
- `sh> awk '/PATTERN/ { print }' knyga.txt`
- `sh> awk '/^XYZ/ { if( $1 > 0 ) print }' coord.dat`

Kiti žinomi *x filtrai

GNU sistemos (tokios kaip Linux) turi daug naudingų filtrų; kai kurie mums reikalingi išvardinti žemiau. Daugiau informacijos rasite `'info coreutils'` (GNU `coreutils` paketo aprašyme).

- `tr` – "translate" – pakeičia vienus simbolius kitais, arba ištrina simbolius (opcija -d):
  - `sh> tr "\r" "\n" < knyga.mac-txt > knyga.linux-txt`
  - `sh> tr -d "\r" < knyga.dos-txt > knyga.linux-txt`
- `wc` – "word count" – suskaičiuoja eilutes, žodžius ir simbolius failuose. Gali suskaičiuoti tik eilutes (opcija -l), tik žodžius (opcija -w) arba tik simbolius (opcija -c):
  - `sh> wc knyga.txt`
  - `sh> wc -w < knyga.txt`
  - Suskaičiuoja .txt failus direktorijoje:
    `sh> ls *.txt | wc -l`
  - Suskaičiuoja atomus PDB failuose:
    `sh> grep '^ATOM ' *.pdb | wc -l`

Perl kaip komandinės eilutės filtras

Perl kalba yra pritaikyta ir trumpoms programėlėms-filtrams komandinėje eilutėje rašyti:

- Kaip sužinoti simbolio kodą? Kad ir taip:
  - `sh> perl -e 'printf "%d\n", ord("A")'`
  - `sh> perl -e 'printf "0x%02X\n", ord("A")'`
- Pritaikyti kiekvienai eilutei reguliarią išraišką ir atspausdinti (`sed` analogas):
  - `sh> perl -pe 's/mano/tavo/g' *.txt`
- Pritaikyti kiekvienai eilutei Perl programą; programa pati atspausdins, ką reikia (`awk` analogas):
  - `sh> perl -ne 'print unless /^\s*$/' *.txt`

Sudėtingesnio uždavinio pavyzdys

Suraskime *anagramas* (t.y. žodžius iš vienodų raidžių) tekstiniuose failuose. Pavyzdžiui, žodžiai “tyras” ir “rytas” yra anagramos.

Perrinkti visas anagramas būtų nepraktiška ($\sim n!$ kombinacijų kiekvienam žodžiui, kur n – raidžių skaičius)

Sprendimo būdas: surūšiuoti (surikiuoti) žodžių raides, o po to naudoti šias eilutes kaip raktus patiems žodžiams rūšiuoti.

Anagramų uždavinio sprendimas

- anagrams.sh:

```
#!/bin/sh
# Find all words that are anagrams in input files
cat $* \
| tr "\r\t" "␣" \
| perl -040 -l012 -ne 'print' \
| perl -CS -lne 'print join("",sort(split(""))), "␣", $_' \
| sort -k1 | uniq \
| perl -lane \
    'sub print_anagrams(@) {
        if( @_ > 1 ) {
            for( @_ ) { print $_->[1] }; print ""
        }
    }
    if( !@p || $p[0][0] eq $F[0] ) {
        unshift(@p,[@F])
    } else {
        print_anagrams( @p );
        @p=([@F])
    }
    END {
        print_anagrams( @p );
    }'
```