10 - Szrk

A szrkrl általában

A szrk a bemenetükre került adatokkal csinálnak valamit, majd a kimenetre küldik.

cat

A cat az egyetlen szr, amely a bemenetre érkez adatokkal semmit nem csinál. Változatlanul írja azokat a képernyre. Mégis ez a parancs – mint azt feljebb láttuk, amellyel a legtöbb dolgot tehetjük. Ezek:

- állomány tartalmának megjelenítése
- állomány létrehozása
- állományok összemásolása
- állományok másolása
- állományhoz fzés

Állomány létrehozása:

```
cat > dolgozok.txt
```

Állomány tartalma:

```
cat dolgozok.txt
```

Állományok összemásolása:

```
cat fajl1 fajl2 fajl3 > egyfajl.txt
```

Állományok másolása:

```
cat < eredeti > masolat
```

A fájl végéhez fzünk:

```
cat >> dolgozok.txt
Szabó Gábor 830000
[Ctrl]+[D]
```

A fájl végéhez fzk:

```
cat >> dolgozok.txt <<VEGE Nagy József 800000 Kis Béla 570000 Jég Lajos 750000 VEGE
```

cat összefoglalva

Állományok létrehozása

```
cat > filenev.txt
tartalom
...
<Ctrl>+<D>
```

Állomány tartalmának megtekintése

```
cat < filenev
```

vagy

```
cat filenev
```

Állományok másolása

```
cat < filenev > filenev2
```

Állományok összemásolása

```
cat file1 file2 file3 > file4
```

colrm

A bemenetre érkez sorokból bizonyos oszlopok törlése

Szintaxis:

```
colrm [startoszlop [ végoszlop ]]
```

A példa kedvéért adott az alábbi tabulátorral tagolt állomány a következ tartalommal:

tanulok.txt001 Alex Brown 85 5 002 Dan Igor 40 3 003 Barton Flex 56 3 004 Rock Brain 78 4

Szeretnénk a nevek oszlopait eltüntetni.

Megoldás:

```
cat tanulok | colrm 9 24
```

Eredmény:

cel.txt001 85 5 002 40 3 003 56 3 004 78 4

Nagyobb példafájl:

tanulok.txt001 Alex Brown 85 5 002 Dan Igor 40 3 003 Barton Flex 56 3 004 Rock Brain 78 4 005 Kis István 72 4 006 Nagy Béla 70 4 007 Fehér Tamás 28 1 008 Kékes Imre 45 4 009 Per Béla 78 4 010 Zöld Elek 78 4 011 Piros Gábor 23 1 012 Fék Lajos 28 4 013 Elmúlt Tibor 78 4 014 Leend Tamás 28 4 015 Pék Gerg 78 4 016 Rob Sándor 18 1 017 Kékedi Imre 70 4 018 Nagy Lajos 85 4

Egyéb használat

```
ifconfig eth0 | colrm 1 10
```

```
cat /var/log/syslog | colrm 1 16 | colrm 30
```

```
cat /var/log/auth.log.1 | colrm 70
```

Utóbbi példában ha szimplán az állomány tartalmát listázom, azok kilógnak a következ sorban. A célom, hogy átláthatóbb sorokat szeretnék, de jelenleg csaka sorok elejére vagyok kíváncsi.

cut

Csak bizonyos oszlopokat szeretnénk megjeleníteni.

Szintaxis:

```
cut [opció] … [fájl] ...
```

Szóközökkel tagolt sorok

Adott az alábbi szóközökkel tagolt állomány:

tanulok2.txt001 Alex Brown 85 5 002 Dan Igor 40 3 003 Barton Flex 56 3 004 Rock Brain 78 4

Megoldás:

```
cat tanulok2.txt | cut -c4-22
```

Az eredmény:

```
Alex Brown
 Dan Igor
 Barton Flex
 Rock Brain
```

Tabulátorral tagolt sorok

tanulok.txt001 Alex Brown 85 5 002 Dan Igor 40 3 003 Barton Flex 56 3 004 Rock Brain 78 4

Megoldás:

```
cut -f2 <  tanulok.txt
```

A futtatás eredménye:

```
cut -f2 < tanulok.txt
Alex Brown
Dan Igor
Barton Flex
Rock Brain
```

Kettsponttal tagolt

tanulok3.txt001:Alex Brown:85:5 002:Dan Igor:40:3 003:Barton Flex:56:3 004:Rock Brain:78:4

Megoldás:

```
cut -f2 -d : tanulok3.txt
```

A -d kapcsoló után szóközzel vagy a nélkül megadjuk az elválasztójelet. A -d kapcsolóval szinte bármit megadhatunk szeparátornak.

Eredmény:

```
cut -f2 -d: tanulok3.txt
Alex Brown
Dan Igor
Barton Flex
Rock Brain
```

Egyszerre több oszlop megjelenítése

```
cut -f 2,3,7 -d : tanulok3.txt
```

Eredmény:

```
cut -f 2,3,4 -d: tanulok3.txt
Alex Brown:85:5
Dan Igor:40:3
Barton Flex:56:3
Rock Brain:78:4
```

mcrypt

A következ parancsok használatához szükséges az mcrypt nev csomag telepítése. A telepítéshez lásd a csomagkezelés részt a „Rendszer” fejezetben.

Titkosítás

Adott az alábbi állomány a következ tartalommal:

nyilt.txtHolnap este érkezem.

Szeretnénk titkosítani az állományt.

Titkosítás:

```
mcrypt nyilt.txt
```

A futtatás eredménye:

```
mcrypt nyilt.txt
Enter the passphrase (maximum of 512 characters)
Please use a combination of upper and lower case letters and numbers.
Enter passphrase:
Enter passphrase:

File nyilt.txt was encrypted.
```

A parancs bekért egy jelszót kétszer, majd létrehozza a következ állományt:

```
nyilt.txt.nc
```

A nyilt.txt.nc szöveg tartalmát ha megnézzük olvashatatlan szöveg lesz. Így már elküldhetjük a titkos üzenetet. Ezek után az üzenet vevjének vissza kell kódolni az üzenetet.

Dekódolás

A visszafejtés a következ paranccsal lehetséges:

```
mdecrypt nyilt.txt.nc
```

A futtatás eredménye:

```
mdecrypt nyilt.txt.nc
Enter passphrase:
File nyilt.txt.nc was decrypted.
```

A parancs bekéri a jelszót, majd elkészíti a nyilt.txt.nc állományt visszakódolva a nyilt.txt nev fájlt.

crypt parancs

A crypt parancsot a mcrypt csomag valósítja meg. Régi unixból ismert parancs, amely a stdin-rl várja a bemenetet, és a stdout-ra ír.

```
crypt < nyilt.txt > titkos.txt
Unix crypt(1) emulation program using mcrypt(1).

Use crypt -h for more help.
Enter the passphrase (maximum of 512 characters)
Please use a combination of upper and lower case letters and numbers.
Enter passphrase:
Enter passphrase:

Stdin was encrypted.
```

Persze ebben a formában is használahtó:

```
crypt > titkos.nc
Unix crypt(1) emulation program using mcrypt(1).

Use crypt -h for more help.
Enter the passphrase (maximum of 512 characters)
Please use a combination of upper and lower case letters and numbers.
Enter passphrase:
Enter passphrase:

Holnap este érkezem.
Stdin was encrypted.
```

Visszafele ugyanígy:

```
crypt < titkos.nc
Unix crypt(1) emulation program using mcrypt(1).

Use crypt -h for more help.
Enter the passphrase (maximum of 512 characters)
Please use a combination of upper and lower case letters and numbers.
Enter passphrase:
Enter passphrase:

Holnap este érkezem.
Stdin was encrypted.
```

grep

Adott mintát tartalmazó sorok megjelenítése

REGULÁRIS KIFEJEZÉSEK

A Unix eszközök közül nagyon sok használ mintaillesztést, s a megadott mintára illeszkedõ adatokon további feldolgozást hajt végre. Vannak olyan parancsok, ahol a felhasználó adja meg a keresendõ mintát, ilyen a grep parancs, másokban rejtve dolgozik a mintakeresõ algoritmus. Reguláris kifejezések használatakor egy komplex mintát adunk meg (ez a reguláris kifejezés), és azt vizsgáljuk, hogy a feldolgozandó adatok melyik része illeszkedik a megadott mintára. A reguláris kifejezések karakterekbõl állnak, ezek közül néhány speciális jelentest hordoz, ezeket metakaraktereknek nevezzük.

A reguláris kifejezéseket meghatározó szabályok:

- Egy egyedülálló karakter, amely nem újsor karakter, és nem a . * [ ] \ ^ $ karakterek egyike, önmagára illeszkedik. Ez azt jelenti, hogy ha a reguláris kifejezés egy a betû, akkor ez a vizsgált szövegben csakis egy darab a karakterre fog illeszkedni.
- A \c karakterpáros, ahol c egy látható karakter, a c karakterre illeszkedik a karakter literális értelmében. Tehet a * illeszkedik a * -ra, a \\ pedig a \ -re.
- A . (pont) karakter egy olyan reguláris kifejezés, amelyik bármelyik (nem újsor) karakterre illeszkedik. Így pl. az ab. minta illeszkedik az a ba, abb, abc, ..., abz, ab0, ... karaktersorokra.
- Ha e egy reguláris kifejezés, akkor e* egy olyan reguláris kifejezés, amely az e reguláris kifejezés 0 vagy többszöri elõfordulását jelzi. Így az a* reguláris kifejezés illeszkedni fog az üres sztringre (hiszen abban zérószor szerepel az a karakter), valamint az a, aa, aaa, ... karakterláncokra.
- A [ ] zárójelbe tett karaktersorozat illeszkedik az abban a pozícióban levõ bármely, a zárójelben felsorolt karakterre. A karakterek felsorolására vonatkozó szabályok:
    - kódjukat tekintve az egymás után következõ karaktereket rovidíteni lehet a kötõjel használatával. Pl. 0-9a-z jelenti az összes számjegyet és az angol abc összes kisbetûjét.
    - a nyitó zárójelet követõ ^ jel a felsorolt karakterek tagadása. Pl. [^0-9] jelenti a bármely nem szám karaktert.
    - a] zárójel csak a felsorolás elsõ tagja lehet
    - a - karaktert a \- karakterpáros jelenti
- Két egymás után írt reguláris kifejezés szintén reguláris kifejezés. pl. [^0-9][0-9] kifejezés a nem szám karaktert követõ szám karaktersorra illeszkedik.
- Két egymástól | jellel elválasztott reguláris kifejezés illeszkedik akar az egyik akar a másik kifejezésre.
- A ^ jel a sor elejére, a $ jel a sor végére illeszti a mintát.

Speciális karakterek
Mivel a reguláris kifejezések mint interpolált füzérek kerülnek kiértékelésre, ezért a következk is használhatók:

\t	tabulátor (HT, TAB)
\n	újsor, soremelés (LF, NL)
\r	kocsivissza karakter (CR)
\f	lapdobás (FF)
\a	cseng (bell) (BEL)
\e	escape (ESC)
\033	oktális kód megadása (mint a PDP-11 nél)
\x1B	hexa karakter
\c[	kontrol karakter
\l	a következkarakter kisbetsre konvertálva
\u	a következ karakter nagybetsre konvertálva
\L	kisbetsek a következ \E-ig
\U	nagybetsek a következ \E-ig
\E	kisbet, nagybet módosítás vége
\Q	reguláris kifejezések meta karakterei elé fordított törtvonalat ra a következ \E-ig

Ezeken kívül a Perl még a következket is definiálja

\w	egy szóban használható karakter (alfanumerikus karakterek és aláhúzás)
\W	minden ami nem \w
\s	szóköz fajta karakter, szóköz, tabulátor, újsor stb.
\S	minden ami nem \s

\d	számjeg
\D	nem számjegy

Ezek a jelölések karakterosztályok megadásában is használhatók, de **nem** intervallum egyik végén.

A következk is használhatók még reguláris kifejezésekben:

\b	szó határának felel meg
\B	nem szó határnak felel meg
\A	csak a füzér elején
\Z	csak a füzér végén
\G	ott folytatja ahol az elz m//g abbahagyta

A \b karakterosztályban használva visszalépés karakternek felel meg. Egyébként mint szóhatár olyan helyet jelöl, ahol az egyik karakter \w és a mellette lev \W. Ennek eldöntésére a füzér elején az els karakter elé és a füzér végén az utolsó karakter utánra egy-egy \W karaktert képzel a rendszer.

A \A és \Z ugyanaz, mint a ^ és a $ azzal a különbséggel, hogy ezek még a m opció használata esetén sem felelnek meg a füzér belsejében lev soremelés karakternek.

Szintaxisa:

```
grep  [ kapcsolók ]  minta    [ fájl ... ]
```

Adott az alábbi belépési napló:

belepesinaplo.txt

2005.10.25 18:30 alex 196.145.43.3

2005.10.26 18:30 joe 216.45.3.2

2005.10.26 18:44 richard 196.145.43.3

2005.10.28 22:31 joe 196.145.43.3

2005.10.29 18:00 alex 196.145.43.3

A futtatás eredménye:

```
grep alex belepesinaplo.txt

2005.10.25 18:30 alex 196.145.43.3
2005.10.29 18:00 alex 196.145.43.3
```

A grep -v azokat jeleníti meg, amelyek nem tartalmazzák a sorokat.

```
grep -v alex belepesinaplo.txt
2005.10.26 18:30 joe 216.45.3.2
2005.10.26 18:44 richard 196.145.43.3
2005.10.28 22:31 joe 196.145.43.3
```

Az alábbi naplóállomány a következ mezket tartalmazza:

dátum	id	felhasználónév	ipcím	letöltött adatmennyiség

naplo.log

2005.12.25 18:30 alex 196.145.43.3 1960 bytes

2005.12.26 18:30 joe 216.45.3.2 1512 bytes

2005.12.26 18:44 richard 196.145.43.3 2005 bytes

2005.12.28 22:31 joe 196.145.43.3 2006 bytes

2005.12.29 18:00 alex 196.145.43.3 2050 bytes

2006.01.02 08:25 joe 195.166.29.5 2008 bytes

2006.01.02 12:20 alex 195.165.1.1 2005 bytes

Feladatunk, hogy listázzuk a 2006-os eseményeket. Ha most a fentebb tárgyalt módon csak ennyit írunk:

```
grep 2006 naplo.log
```

úgy azok a sorok is megjelennek ahol a letöltött byte-ok száma 2006. Ezért ez nekünk nem jó. Jeleznünk kell, hogy csak azokat a sorokat szeretnénk megjeleníteni, ahol a 2006 a sor elején szerepel. Ehhez a ^ karaktert használjuk:

```
grep ^2006 naplo.log
```

Fájlnevek megjelenítése tartalom alapján: Például keressük az aktuális könyvtárban, az összes fájl között, azokat a fájlokat, amelyek tartalmazzák a VGA szót, megjeleníteni azonban csak a fájlneveket szeretnénk:

```
grep -l VGA *
```

A karakterosztályok használata

naplo.log

2005.12.25 18:30 alex 196.145.43.3 1960 bytes

2005.12.26 18:30 joe 216.45.3.2 1512 bytes

2005.12.26 18:44 richard 196.145.43.3 2005 bytes

2005.12.28 22:31 joe 196.145.43.3 2006 bytes

2005.12.29 18:00 alex 196.145.43.3 2050 bytes

2006.01.02 08:25 joe 195.166.29.5 2008 bytes

2006.01.02 12:20 alex 195.165.1.1 2005 bytes

2006.01.03 07:02 mari 195.165.2.45 2007 bytes

A fenti állományban azokat a sorokat keressük, ahol a letöltés 2005, 2006, 2007 vagy 2008 van.

```
grep "200[5678] bytes" naplo.log
```

A negyedik karakter csak a szögletes zárójelben ”[ ]„ megadott négy karakter egyikére illeszkedik. Az 5,6,7 vagy 8-as karakterek valamelyikére. A szám után kötelezen egy szóköznek, majd utána bytes karakterek megköveteltek.

A szögletes zárójel lehetvé teszi intervallumok megadását [5-8]:

```
grep "200[5-8] bytes" naplo.log
```

Az intervallumok megadására használhatók betk is:

```
[a-z]
```

Az elbbi minta például az angol ábécé összes kisbetjére illeszkedik. A nagybetk:

```
[A-Z]
```

Természetesen megadhatók kisebb intervallumok is:

```
[c-k]
```

Üres sorok törlése

```
grep -v "^$" filename
```

```
grep . filename
```

```
grep -v '^[[:space:]]*$' filename
```

Hashmárk jeles sorok kihagyása

```
grep -v '\#' filename
```

Fájlok keresése tartalom alapján

```
grep -H -r “Rewrite” /etc/apache2/
```

```
man -L en grep
```

grep gyakorlat

Adott az alábbi naplóállomány részlet (/var/log/syslog):

```
Feb 21 18:32:12 server postfix: Connection,
Feb 21 18:32:12 server postfix: LOGIN, user=test@server.hu,
Feb 21 18:32:46 server postix: LOGOUT, user=test@server.hu
Feb 21 18:32:12 server pop3d: Connection,
Feb 21 18:32:12 server pop3d: LOGIN, user=test@server.hu,
Feb 21 18:32:46 server pop3d: LOGOUT, user=test@server.hu
```

Csak a pop3d-t tartalmazhó sorokat szeretnék megjeleníteni.

```
grep pop3d /var/log/syslog
```

Adott egy konfigurációs állomány (dspam.conf) ami tele van megjegyzésekkel. Szeretném kiszrni a megjegyzés sorokat és csak a beállításokat szeretném:

```
#
# DSPAM Home: Specifies the base directory to be used for DSPAM storage
#
Home /var/spool/dspam
```

Ekkor:

```
grep '^[^#]' dspam.conf
```

Fájl keresése tartalom alapján

```
grep -lir "keresett szöveg" /utvonal/konyvtar
```

egrep

Nem egyezik meg a grep -E használatával, mert annál több reguláris kifejezést ismer ha ezt használjuk.

Ehhez hasonló reguláris kifejezések is használhatók:

```
+,?, | és ()
```

fgrep

Fix vagy fast grep rövidítése. Megegyezik a grep -F használatával

paste

Adatoszlopok összemásolása

Állományok oszlopainak összemásolása egy fájlba, egymás mellé

Szintaxis:

```
paste   [ kapcsolók ... ]  [ fájl ... ]
```

Van 4 darab fájl:

- sroszam
- nev
- szuletesnap
- telefonszam

sorszam001 002 003

nevAlex Brown Joe

szuletesnap1985.07.11 1987.01.12 1999.12.22

telefonszam(1) 335-3345 (37) 332-328 (48) 183-981

Szeretnénk egyetlen fájlban egymás mellé összemásolni.

Megoldás:

```
paste sorszam nev szuletesnap telefon  > info.txt
```

Eredmény:

info.txt001 Alex 1987.07.11 (1)335-3345 002 Brown 1987.01.12 (37) 332-328 003 Joe 1999.12.22 (48) 143-981

Tagolás szóközökkel

Megoldása

```
paste -d ' ' sorszam nev szuletesnap telefon  > info
```

Eredmény:

info001 Alex 1987.07.11 (1)335-3345 002 Brown 1987.01.12 (37) 332-328 003 Joe 1999.12.22 (48) 143-981

Tagoláshoz persze bármit megadhatunk pl.:

```
-d ':'
```

Táblázat oszlopainak cseréje

Adott egy dolgozok nev állomány, amiben oszlopokban szeretnénk megcserélni.

dolgozokNév Fizetés Beosztás Alex 130000 fest Brown 150000 asztalos Joe 170000 villanyszerel

Megoldás:

```
cut -f1 dolgozok  >  nev
cut -f2 dolgozok >  fizetes
cut -f3 dolgozok >  beosztas

paste nev beosztas fizetes >  dolgozok2

more dolgozok2

rm nev beosztas fizetes
```

sort

A bemenetre érkez adatokat rendezi.

A sort parancsot Adatok rendezése és összemásolása

Ha a bemenet egy másik programtól jön csvezetéken:

```
cat filenev.txt | sort
```

Ha a program bemenete egy állomány:

```
sort < filenev.txt
```

Fordított rendezés

```
sort -r nevek.txt
sort --reverse nevek.txt
```

Kimeneti fájl megadása

```
sort -o ujfajl.txt  nevek.txt
```

Átirányítással:

```
sort nevek.txt > ujfajl.txt
```

Ha így használjuk:

```
sort nevek.txt > nevek.txt
```

A nevek.txt tartalma elveszett, mert a kimeneti fájl elkészítése megtörténik annak olvasása eltt.

Itt lehet ugyanaz a név:

```
sort -o nevek.txt  nevek.txt
```

Egy fájlba

```
sort nevek1 nevek2 nevek3 > osszes
```

Egy fájlba rendezi a 3 fájl tartalmát

A gyümölcs fájl tartalma:

gyumolcs.txtkorte alma szilva barack

Eredmény:

```
sort gyumolcs.txt
alma
barack
korte
szilva
```

Rendezés adott mez szerint

gy.txt1

körte 2

alma 3

szilva 5

Második oszlop szerint szeretnénk rendezni:

```
sort -k +2 < gy.txt
```

-k oszlopszám

rev

Egy sort megfordít

A rev parancsot karakterek sorrendjének megfordítása használjuk.

Szintaxis:

```
rev [fájl …]
```

A bemenet lehet a STDIN is.

Legszemléletesebben egy egyszer állományon láthatjuk a mködését. Az adat.txt állomány tartalma legyen a következ:

adat.txt

12345

abcde

AxAxA

Futtatás:

```
rev adat.txt
54321
edcba
AxAxA
```