Bash

Hol tudunk gyakorolni?

- A legtöbb rendszer megfelel, ahol a Bash parancssor elérhető
  - A Glomer – ahol a házi feladatokat és a beugrót kell majd megoldani
  - Saját Linux rendszer – más tárgyakhoz is ajánlott
  - Virtuális gép – ha „elrontjuk”, nem okozunk kárt a gazda rendszerben
  - Live rendszer – ezzel akár egy kölcsön gépen is nyomtalanul gyakorolhatunk

További segítség

- https://linuxconfig.org/bash-scripting-tutorial-for-beginners
- https://www.perltutorial.org
- https://www.tutorialspoint.com/perl/index.htm

- (Felzárkóztatás)

Interaktív bash

Belépés egy Unix rendszerbe

- Általában
  - Loginnév + jelszó párossal történik
  - Egy időben több felhasználó is beléphet
  - Egyszerre több „példányban” is beléphetünk
- A felhasználók egymástól el vannak választva
  - Pl. külön „home” könyvtár
- Grafikus, vagy szöveges felületet kapunk. A grafikus felületen indított terminál egyenértékű a valódi szöveges terminállal

Parancsok bevitele

- A parancsokat a prompt után tudjuk beírni
  - A prompt végét $ vagy # jel jelzi
  - A beírt parancsok a szokásos módon bal és jobb nyíl, backspace, delete, stb. billentyűkkel szerkeszthetők.
  - A fel és le nyilakkal az előzőleg kiadott parancsokat hozhatjuk elő.
  - A Ctrl-R kombinációval keresni tudunk régebbi parancsok között, ha emlékszünk a parancs egy részére
  - A history parancs sorszámozva kiírja az eddigi parancsokat
    - !12 Végrehajtja a 12-es számú parancsot
    - history -c Törli a historyt.

Egyéb hasznos billentyűparancsok

- A Ctrl-C lelövi a jelenleg futó programot. (Nem mindig működik.)
- A Ctrl-L törli a képernyőt. Az éppen szerkesztés alatt álló parancs megmarad.
- A tabulátor megpróbálja kiegészíteni a parancsot
- Ha nem tudja teljesen kiegészíteni, akkor egy újabb tabulátort nyomva megmutatja a lehetőségeket.

A Unix fájlrendszer

- Fa struktúra egy közös gyökér könyvárral (nincsenek meghajtó betűjelek)
- Minden entitás fájl. Több fájl típus van:
  - Hagyományos (regular) fájl
  - Könyvtár (speciális fájl)
  - Egyéb speciális fájlok
    - Hard link
    - Soft link
    - Karakteres és blokk eszköz
    - Fifo

Elérési utak

- A gyökér könyvtár jele: /
- Az egymásból nyíló könyvtárakat is / jel választja el.
  - Így megadhatók abszolút útvonalak:
    - /bin
    - /usr/bin/perl
    - /home/antiemes/proba/olvassel.txt
  - Illetve relatív útvonalak:
    - proba/olvassel.txt

Speciális elérési utak

- . (pont): Aktuális könyvtár
- .. (két pont): Egy szinttel feljebb levő könyvtár
- ~ (hullámvonal): Az aktuális felhasználó home könyvtára
- Például:
  - ./elso.sh Az aktuális könyvtárban levő elso.sh fájl
  - ../doksi.txt Az egy szinttel feljebb levő doksi.txt fájl
  - ../../kep/szep.jpg A két szinttel feljebb levő kep könyvtárban levő szep.jpg

Mozgás a könyvtárak között

- A cd parancssal történik:
  - cd elérési_út
- Például:
  - cd / visszavisz a főkönyvtárba
  - cd .. egy szinttel feljebb visz
  - cd /ez/az/amaz belép a gyökérkönyvtárból indulva az ez/az/amaz-ba
  - cd ez/az/amaz belép az aktuális könyvtárból indulva az ez/az/amaz-ba
  - cd ~/ez/az/amaz belép a saját home könyvtárunkból indulva az ez/az/amaz-ba

Az aktuális könyvtár

- A pwd (print working directory) paranccsal kérdezhető le.
- Az aktuális könyvtár általában a promptban is megjelenik.
  - [antiemes] [Flow] [/mnt/proj/OS_new] [$]
  - Itt a /mnt/proj/OS_new az aktuális könyvtár

Könyvtárak tartalmának listázása

- Az ls parancssal történik.
- Az ls parancs rengeteg paramétert tud fogadni, amivel meg tudjuk mondani, hogy:
  - Mit listázzon: elérési utakkal
  - Hogyan listázzon: kapcsolókkal
- Ha nem adunk meg paramétert, akkor egy egyszerű listát kapunk az aktuális könyvtárról
- Egy, vagy több elérési úttal (lehetnek fájl, vagy könyvtárnevek is) azokat listázza

Az ls parancs kapcsolói – I.

- -l (long) hosszú lista, minden adattal
- -a (all) rejtett bejegyzéseket is listázza
- -h (human) „emberi” mértékegységek a fájl mérethez
- -1 csak egy oszlopot használ
- -R (recursive) rekurzívan listáz
- -d (direcory) a megadott könyvtárnak nem a tartalmát, hanem magát a könyvtár bejegyzést írja ki
- Ezeken kívül még rengeteg egyéb lehetséges opció

Az ls parancs kapcsolói – II.

- A kapcsolók keverhetők is:
  - -a -l Hosszú lista, rejtettek is
  - -al Így is írhatjuk
  - -alh Ugyanaz, csak emberi mértékegységekkel
- A kapcsoló mellett megadhatunk egy, vagy több elérési utat is:
  - ls -l / A gyökérkönyvtárról hosszú lista
  - ls -R /home/antiemes/stuff A megadott könyvtárról rekurzív lista

Az ls hosszú listája

- Egy példa:
  - total 16
  - -rw-r--r-- 1 antiemes users 2650 Feb 6 16:45 BlinkAppC.nc
  - -rw-r--r-- 1 antiemes users 2924 Feb 6 16:45 BlinkC.nc
  - drwxr-xr-x 3 antiemes users   60 Feb 6 16:45 build
  - -rw-r--r-- 1 antiemes users   42 Feb 6 16:45 Makefile
  - -rw-r--r-- 1 antiemes users 678 Feb 6 16:45 README.txt
- Első sor: az összes elfoglalt hely blokkban
- Ezt követően 7 oszlop

Az ls -l hét oszlopa

- Típus és jogosultságok (később lesz róla szó)
- Link counter (később lesz róla szó)
- Tulajdonos
- Tulajdonos csoport
- Méret byte-ban (átállítható)
- Dátum (az utolsó módosítás az alapértelmezett)
- Név

Bejegyzés típusa

- Egy karakter a típus, majd 3 hármas csoportban a jogok
- Típusok:
  - - sima fájl
  - d könyvtár
  - l soft link (erről majd később)
  - c karakteres eszköz *
  - b blokk eszköz *
  - c socket *
  - p fifo *

*: csak a teljesség kedvéért

Jogosultságok

- 3db hármas csoport: tulajdonos (u), csoport (g) és mások (o) jogai
  - r: olvasási jog
  - w: írási jog
  - x: végrehajtási jog
  - -: az adott jog nem engedélyezett
  - Egyéb betűk is lehetségesek, ezekkel nem foglalkozunk
- A végrehajtási jog könyvtáraknál speciális jelentésű: belépési jogok jelent.

Belső dokumentáció

- man (manual) parancs
  - man [fejezet] parancs_vagy_tema
  - A man kimenetét a less (vagy a more) program teszi kereshetővé, oda-vissza görgethetővé
- A programok után írt -h, -help, --help kapcsolók is sok esetben szolgálnak információval.
- A man oldalakat az Interneten is megtaláljuk
- Például
  - man ls
  - man man
  - man 7 regex

Könyvtár létrehozása (mkdir)

- mkdir elérési_út
  - Ha az elérési út csak egy könyvtár, akkor az aktuális könyvtárban hozza létre.
  - Ha összetett elérési utat adunk meg, akkor csak akkor hozza létre, ha az útvonal „eleje” már létezik. -p kapcsolóval az összes hiányzó könyvtárt létrehozza.
  - Több könyvtár is megadható egymás után

Fájl létrehozása tartalom nélkül (touch)

- touch név1 [név2 [név3 …]]
- A nem létező fájlok üres tartalommal létrejönnek
- Ha nem adunk meg elérési utat, akkor az aktuális könyvtárba kerülnek, egyébként oda, ahova megadtuk
- A már létező fájlokat nem írja felül, de az utolsó módosítás dátuma frissül

Másolás (cp) – I.

- cp [kapcsolók] forrás cél
  - Sokféle működési eset van, de a logika hasonló
  - A forrást másolja a cél helyre
  - Forrás: létező fájl, cél: nem létező fájl
    - Az eddig nem létező fájl létrejön és megkapja a régi tartalmát.
    - Megadhatunk elérési utakat is. Ha nem adunk meg, akkor az alapértelmezett az aktuális könyvtár.
  - Forrás: létező fájl, cél: létező fájl
    - Mint az előző eset, de a cél fájl felülíródik.
  - Forrás: létező fájl(ok), cél: létező könyvtár
    - A forrás fájlokat átmásolja a cél könyvtárba.
    - Ha már volt ott valami, azt felülírja.

Másolás (cp) – II.

- Másolhatunk könyvtárakat is, ilyenkor a -r kapcsolót kell megadni.
  - r: recursive
  - A könyvtárak tartalmát is másolja
- A forrás megjelölésénél használhatunk Joker karaktereket is (shell mintákat)
  - cp *.txt konyvtar — A .txt-re végződőeket másolja
  - cp *ab* konyvtar — Azokat másolja, aminek a nevében van ab
  - cp vala* konyvtar — Azokat másolja, ami vala-val kezdődik
  - cp *a*a* konyvtar — Azokat másolja, amiknek a nevében van 2db „a” betű
  - De itt nem az történik, amit gondolunk!
  - A cp itt is egy fájl (vagy könyvtár) listát kap, ugyanúgy, mint amikor mi kézzel megadtuk, hogy mit másoljon

Szöveg kiírása (echo) – I.

- echo szoveg
  - szoveg ← A szöveg simán kiíródik
- echo ez tobb szavas
  - ez tobb szavas ← A több szavas szöveg is kiíródik
- echo itt sok space van
  - itt sok space van ← Csak egy space-t ír ki!
- Magyarázat
  - Ahány szót adunk meg, az echo annyi paramétert kap.
  - A paramétereket szóközzel elválasztva írja ki.
- Megoldás: idézőjel
  - Echo ”itt sok space van” Egyben fogja tartani a szavakat, az echo egy paramétert kap, így megmarad a sok space.

Szöveg kiírása (echo) – II.

- Az echo automatikusan sort emel a szöveg végén. A soremelés -n kapcsolóval mellőzhető.
- Escape szekvenciák értelmezése -e kapcsolóval lehetséges.
  - echo -e ”Ezt\tszepen\ttabulalva\tirja\tki.”
  - echo -e ”Minden\nszot\nuj\nsorba\nteszunk.”

Shell minták...

- echo *
  - Nem egy * karaktert fog kiírni, hanem az ott szereplő fájlok (és könyvtárak) listáját
  - A shell a * helyére behelyettesíti azokat fájlokat és könyvtárakat, amik illenek a mintára.
  - A cp és társai esetében is ugyanúgy működik. Nem shell mintát kapnak, hanem fájl listát
- A feloldás ellen escape-eléssel lehet védekezni
  - echo ”*”
  - echo '*'
  - echo *

Törlés (rm)

- rm [kapcsolók] fájl_és_könyvtár_lista
  - A remove szóból jön.
  - A lista ugyanúgy megadható abszolút, vagy relatív elérési úttal is. Egyéb esetben a jelenlegi könyvtárból töröl.
  - Shell minta is megadható.
  - Könyvtár a -r (recursive) kapcsolóval törölhető. Ekkor a könyvtár teljes tartalma is törlődik.
  - A csak olvasható fájlok esetén rákérdez, egyébként kérdés nélkül töröl.
    - -f megadása esetén soha nincs kérdés
    - -i esetén mindig van kérdés

Könyvtár törlése (rmdir)

- rmdir könyvtár [könyvtár2 [könyvtár3 ...]]
  - Csak üres könyvtárt töröl
  - Itt is használhatunk relatív, vagy abszolút elérési utakat, illetve shell mintákat
    - Pl. az rmdir * minden üres könyvtárt töröl. A többi tartalmat meghagyja.

Átnevezés, áthelyetés (mv)

- mv forrásfájl célfájl
  - Ha a forrás és a cél ugyanabban a könyvtárban van, akkor átnevezi a fájlt.
  - Ha a két könyvtár eltér, akkor áthelyezés is történik.
  - Ha a célfájl létezik, akkor felülírja azt.
- mv forrásfájlok és könyvtárak célkönyvtár
  - Ebben az esetben a megadott fájlok és könyvtárak a cél könyvtárba kerülnek.
  - Shell mintákat ebben az esetben is használhatunk.

Tartalom kiírása (cat)

- cat [fájl1 [fájl2 [fájl3 ...]]]
- A concatenate szóból jön. (Állítólag...)
- A fájlok tartalmát egymás után kiírja.
- Még ennek az egyszerű programnak is számos kapcsolója van (pl. meg tudja számozni a sorokat...)
- Paraméterek nélkül a standard bemenetről veszi az inputot.
  - A bevitel végét Ctrl-D-vel jelezzük.

Tartalom lapozott kiírása (less, more)

- less fájl
  - A man-hoz hasonlóan a fájl fel-le görgethető, kereshető, stb.
  - A man is ezt a programot használja.
- more fájl
  - A less-hez hasonló.
  - Kicsit butább.

Kimenet átirányítása

- A legtöbb program a képernyőre kiír valamit. Ezt át tudjuk irányítani egy fájlba is, a > jellel.
  - ls -l > fajl
  - ls -R > fajl Mindkét esetben a képernyő helyett a fajl fájlba kerül a kimenet
  - Ha volt már ott fájl, akkor felülíródik!
- A kimenetet hozzá is tudjuk fűzni egy fájlhoz a >> jellel
  - cat a*txt >> asok Az ”a”-val kezdődő, txt-re végződő fájlok tartalmát az ”asok” fájlhoz fűzi hozzá
- cat > program.sh
  - Be tudjuk gépelni a tartalmát (legvégső megoldás).
  - Be tudunk illeszteni egy szöveget a vágólapról (nagyon hasznos!).

Bemenet átirányítása

- Sok program a billentyűzetről vár inputot. A standard bemenetet is átirányíthatjuk úgy, hogy a program a billentyűzet helyett egy fájlból vegye az adatokat.
- program < fájl
- Hasznos alkalmazás amikor egy saját programot tesztelünk.
  - ./pelda < bemenet.txt

Fájlok linkelése – I.

- Soft link
  - Egy olyan speciális fájl, ami egy másik fájl elérési útját tartalmazza.
  - Ha hivatkozunk erre a fájlra (vagy könyvtárra), akkor (általában) egyenértékű a másikra való hivatkozással
    - Kivétel pl. az átnevezés, törlés
  - Létrehozása: ln -s létező_fájl [cél könyvtár vagy cél fájl]
    - Ha a cél egy fájl, akkor azon a néven jön létre a link
    - Ha a cél egy könyvtár, akkor a régi néven jön létre a kijelölt könyvtárban
  - Mivel a link csak egy hivatkozás, az eredeti fájl törlése esetén a tartalom el fog veszni.

Fájlok linkelése – II.

- Hard link
  - Ugyanahhoz a tartalomhoz két fájl bejegyzés
  - ln létező_fájl új_fájl
  - Ha az egyiket töröljük, a tartalom megmarad.
  - A link counter mutatja, hogy hány hivatkozás van az adott tartalomra.
  - Csak egy fájlrendszeren belül hozhatunk létre hard linket.
  - Könyvtárra nem lehet hard linket létrehozni.

Jogosultságok módosítása – I.

- chmod módosítás fájl1 [fájl2 [fájl3 ...]]
- A módosítás lehet kinek[+-]mit alakú
  - Kinek:
    - u: user (tulajdonos)
    - g: group (csoport)
    - o: others (mások)
    - a: all (mindenki)
  - Mit:
    - r: read (olvasási jog)
    - w: write (írási jog)
    - x: execute (végrehajtási/belépési jog)
  - +: megad, -: elvesz

Jogosultságok módosítása – II.

- Például:
  - u+x A tulajdonosnak futtatási jogot adunk
  - ug+w A tulajdonosnak és a csoportnak írási jogot adunk
  - a+x Mindenkinek adunk futtatási jogot
  - go-rwx Mindenkitől elveszünk minden jogot, kivéve magunktól.
- A jogosultságok nem csak változtathatók, hanem egy lépésben be is állíthatók
  - A 3db hármas csoportot 3db oktális számmal jelöljük
  - x: 1-esek, w: 2-esek, r: 4-esek.
  - chmod 640 doksi.txt A tulajdonos írhatja és olvashatja, a csoport olvashatja, másoknak nincs semmilyen joga.
  - chmod 777 pelda.sh Mindenki minden műveletet elvégezhet a fájllal

Tulajdonos megváltoztatása

- chown uj_tulaj fájl1 [fájl2 [fájl3 ...]]
  - A tulajdonos a továbbiakban az uj_tulaj lesz.
- chown uj_tulaj:uj_csoport fájl1 [fájl2 [fájl3 ...]]
  - A tulajdonos az uj_tulaj, a csoport az uj_csoport lesz.
- chgrp uj_csoport fájl1 [fájl2 [fájl3 ...]]
  - A csoport az uj_csoport lesz.
- A -R kapcsolóval rekurzívan működik, tehát az alkönyvtárakba is belemegy.
- Ehhez a művelethez rendszergazdai jogosultság kell!

Fájl eleje és vége

- head [-n szám] fájl1 [fájl2 [fájl3 ...]]
  - A fájl első „szám” sorát írja ki. Ha nincs megadva ez a paraméter, akkor 10 sor az alapértelmezett.
  - Kiírja a fájl nevét is (-q kapcsolóval eltüntethető).
- tail [-n szám] fájl1 [fájl2 [fájl3 ...]]
  - Hasonló, csak az utolsó „szám” sort fogja kiírni.
  - Shell minta mindkét programnál használható.
- Pl.:
  - head -n 8 leiras.txt — A leiras.txt első 8 sorának kiírása.
  - tail readme — A readme utolsó 10 sorának kiírása.
- Ha nem adunk meg fájlt, akkor a standard inputról olvas.

A pipe – I.

- A standard outputra író és a standard inputról olvasó programok összeköthetőek.
  - ls -l | tail -n 5 Az ls -l által adott tartalomnak az utolsó 5 sorát adja vissza.
  - cat olvassel.txt | head -n 5 Az olvassel.txt első 5 sorát adja vissza.
    - Majdnem ugyanazt csinálja, mint a head -n 5 readme.txt. De csak majdnem!
    - A pipe használata esetén a head nem tudja, hogy honnan jönnek az adatok, így a fájlnevet sem. Így nem tudja kiírni.

A pipe – II.

- Több programot is össze lehet fűzni:
  - cat pelda.txt | head -n 20 | tail -n 1
    - Először vesszük a pelda.txt első 20 sorát.
    - Majd ennek a listának az utolsó sorát.
    - Az eredmény a pelda.txt 20. sora.
  - ls -l | head -n 6 | tail -n 1
    - Hasonló, mint az előző, csak az ls kimeletével operál.
    - Az ls -l kimenetének 6. sorát adja vissza.

Sorok, szavak, karakterek száma (wc)

- wc: word count (de a sorokat és a karaktereket is számolja)
- Kapcsolókkal mondjuk meg, hogy mit számoljon
  - -l sorok száma
  - -w szavak száma
  - -c karakterek száma
  - A kapcsolók kombinálhatók is (pl. -wl a szavakat és a sorokat számolja meg).
  - Az alapértelmezett működés mindhárom szám kiírása.

wc – II.

- Többféle működési mód
  - Pipe
    - ls -1 | wc -l (fájl/könyvtár bejegyések számlálása)
    - cat pelda.txt | wc -w (a pelda.txt szavainak számlálása)
    - A program csak a számot, vagy számokat írja ki, a fájl nevét (ha van) nem.
  - A paraméter 1 fájl
    - wc -c pelda.txt (a pelda.txt sorainak számlálása)
    - A szám(ok) után a program a fájlnevet is kiírja.
  - A paraméter több fájl
    - Egyenként minden fájl sorainak (szavainak, ...) számát kiírja, utána az adott fájl nevét is.
    - A lista végén egy összesített számot is kapunk.
  - Shell minták természetesen itt is használhatóak.

Mezők vágása (cut) – I.

- Karakterenként, vagy mezőnként tudunk vele vágni
- Karakterenként:
  - cut -c kezdő-vég A [kezdő, vég] intervallum által meghatározott karaktert adja vissza minden sorban
  - cut -c sorszám1,sorszám2,sorszám3 A kijelölt 3 karaktert adja vissza
  - cut -c sorszám1,kezdő1-vég1,sorszám2 A fenti kettő kombinálható is (Egy kiválasztott karakter, egy intervallum, majd még egy kiválasztott karakter)
- Pl.:
  - ls -l | cut -c 2-10 A jogokat adja vissza.

Mezők vágása (cut) – II.

- Mezőnként
  - cat /etc/passwd | cut -d: -f 1,5
  - A passwd fájl kettőspontokkal határolt mezőket tartalmaz. Ebből az 1. és 5. mezőt írja ki.
  - cat lista_nagy.txt | cut -f 1,3
  - A lista_nagy fájl tabulátorokkal határolt mezőket tartalmaz, ebből az 1. és 3. mezőt írja ki a parancs. (Ha nem adunk meg mezőhatárolót, akkor a tabulátor az alapértelmezett.)

Rendezés (sort)

- A sort paranccsal sorokat tudunk rendezni
  - cat nevek.txt | sort — ABC sorrendbe rendez
  - sort nevek.txt — Pipe nélkül is használható.
- Megadhatjuk, hogy melyik mező szerint rendezzen
  - ls -l | sort -k 3 — A tulajdonos szerint rendez.
- Numerikus rendezés
  - ls -l | sort -n -k 5 — Méret szerint rendez.
  - ls -l | sort -n -k 5 -r — Visszafelé is rendezhetünk.
- Egy összetettebb példa
  - ls -l | sort -k 5 -n | tail -n 1 — A leghosszabb fájl adatai.

Egyező sorok törlése (uniq)

- Ha többször szerepel a bemenetben ugyanaz a sor egymás után, akkor csak egyet hagy meg.
- Rendezés után hasznos
  - cat nevek.txt | sort | uniq
  - Ebben az esetben a sort -u is használható (ugyanazt csinálja)

Szűrés (grep) – I.

- Azokat a sorokat kapjuk vissza, amik illeszkednek egy reguláris kifejezésre. (A reguláris kifejezések egy későbbi dián vannak részletezve.)
- A találatok kiírása nélkül is megtudhatjuk, hogy a bemenetben szerepel-e az adott minta, vagy nem.
- A keresés kis-nagy betű érzékeny. Ez a -i kapcsolóval kikapcsolható.
- Ha a bemenet pipe:
  - parancs | grep kapcsolók minta
- Ha a bemenet fájl:
  - grep kapcsolók minta fájl

Szűrés (grep) – II.

- Szintaxisok
  - Alap reguláris kifejezés szintaxis
    - A ?, +, {, |, (, és) karakterek normál karakterként működnek. Metakarakterként a \?, \+, stb. használható.
  - Extended szintaxis
    - Egy kicsit többet tud
    - A ?, +, stb. szerepe fordított.
    - grep -E, vagy egrep parancs
  - Perl szintaxis
    - Még egy kicsivel többet tud (pl. \w, \d, \s)
    - Az újabb grep implementációkban a -P kapcsolóval használható

Szűrés (grep) – III.

- További hasznos kapcsolók
  - -v Inverz keresés. Azok a sorok maradnak meg, amelyekre nem illeszkedik a minta.
  - -e minta A grep a mintát veszi reguláris kifejezésnek. (Nem fogja pl. azt hinni, hogy az egy kapcsoló, ha pl. a minta - jellel kezdődik)
  - -A szám A talált sor után még szám db sor kiíródik
  - -B szám A talált sor előtt még szám db sor kiíródik
  - -q Nem íródik ki semmi, csak a grep visszatérési értéke lesz beállítva. (Az if paranccsal együtt lesz majd hasznos.)
- Megkapott fájlnevek
  - A grep pipe-pal használva, vagy egy megadott fájl esetén a fájlnevet nem írja ki. Több fájlnévnél minden sor elején megjelenik a fájlnév is. -H kapcsolóval kikényszeríthető, -h kapcsolóval tiltható a fájlnevek kiírása.

Processek és jobok – I.

- A /proc alatt egy virtuális fájlrendszer található.
  - Számmal jelzett könyvtárak. Minden processhez (folyamathoz) tartozik egy, a szám a folyamat azonosító (PID). Itt az adott folyamattal kapcsolatos információkat találunk:
    - Nyitott fájlok
    - Memória és CPU idő statisztika
    - Tulajdonos
    - Egyéb állapot információk
  - Egyéb könyvtárak és fájlok, ahol egyéb, rendszerszintű dolgokat lehet lekérdezni, vagy beállítani.

Processek és jobok – II.

- A ps parancs a folyamatokról tud listát és információkat adni
  - A /proc alapján dolgozik
  - Kapcsolók:
    - aux (Ez 3 kapcsoló egymás után írva.) Minden folyamatot listáz, minden információval.
    - w, ww Széles formátumot használ, kiírja az egész folyamatnevet és a paramétereket.
    - f Fa struktúrát jelenít meg a szülő-gyerek leszármazási reláció alapján
  - A listában megjelenik a folyamat tulajdonosa, PID-je, CPU és memória használata, állapota, neve és még néhány egyéb információ.

Processek és jobok – III.

- kill parancs
  - kill [-signal_szám | -signal_név] PID
  - Egy üzenetet küld a folyamatnak, amit a signal_szám határoz meg. Mindegyik számhoz egy név is tartozik. Például:
    - 15 (TERM) A programot megkérjük, hogy lépjen ki. (Nem kötelező kilépnie.)
    - 9 (KILL) A programtól elveszünk minden további processzoridőt és kiléptetjük.
    - 17, 19, 23 (STOP) A programot egy időre felfüggesztjük.
    - 19, 18, 25 (CONT) A programot aktiváljuk.
  - Killall [-signal_szám | -signal_név] folyamatnév
    - PID helyet folyamatnevet is használhatunk, ekkor a parancs kikeresi a megfelelő folyamatokat és mindegyiknek elküldi a kijelölt signal-t.

Processek és jobok – IV.

- A terminálban egyszerre több programot is futtathatunk, külön jobként. Ezek közül mindig legfeljebb egy van előtérben.
  - jobs Listázza az aktuális jobokat
  - bg [szám] Az aktuális, vagy egy kijelölt számú jobot háttérbe helyezi.
  - fg [szám] Az aktuális, vagy egy kijelölt számú jobot előtérbe helyezi.
  - kill %szám A PID helyett a job azonosítót is megadhatjuk, % jel után.

Processek és jobok – V.

- Egy programot a háttérben is el tudunk indítani, pl:
  - xterm &
- Az aktív folyamatot meg tudjuk állítani a Ctrl-Z paranccsal
  - Ezután az fg, vagy a bg paranccsal előtérbe, vagy háttérbe tudjuk helyezni.
  - Van olyan program, ami ezt a fajta működést letiltja.

Keresés (find) – I.

- Fájlok és könyvtárak keresése
  - Név alapján (lehet shell minta, vagy reguláris kifejezés)
  - Típus alapján (fájl, könyvtár)
  - Méret és egyéb tulajdonságok alapján
- A találatok
  - Szimpla listázása
  - Valamilyen parancs végrehajtása rajtuk

Keresés (find) – II.

- Szintaxis
  - find könyvtár kapcsolók kifejezés
- Könyvtár
  - Itt fog keresni.
  - A lista elején ez a könyvtár jelenik meg (kivéve a ~).
- Kapcsolók
  - -name, -iname Kereső minta, shell minta (a -iname nem case sensitive).
  - -regex, -iregex Kereső minta, regex alakban (a -iregex nem case sensitive).
  - -type f, -type d Fájl típusa (f: sima fájl, d: könyvtár, default: mindkettőre keres)
  - -size A fájl mérete (legalább, legfeljebb) mekkora
  - -exec Minden találaton egy parancs végrehajtása (erről később)

Keresés (find) – III.

- Útvonalak és típusok
  - find Paraméterek nélkül az aktuális könyvtárban keres, sima fájlt és könyvtárt egyaránt, minden találatot kiír.
  - find . A . paramétert megadva ugyanezt teszi.
  - find proba A proba könyvtárban keres.
  - find proba -type f

    A proba könyvtárban keres sima fájlokat.
  - find proba -type d

    A proba könyvtárba keres könyvtárakat.

Keresés (find) – IV.

- Shell minták
  - find -name ”*.txt”
    A .txt-re végződő bejegyzések keresése. A kereső mintát célszerű idézőjelbe tenni.
  - find -name ”*.sh” -type f A .sh-ra végződő sima fájlok keresése
  - find -name ”fe*t” -type f A fe-vel kezdődő, t-re végződő fájlok keresése
  - find -name ”*a*m*” -type f Azok a fájlok, amik nevében van a és m betű (ebben a sorrendben).
  - find -name ”*[xyz]*” -type f Azok a fájlok, amik nevében van x, y, vagy z

Keresés (find) – V.

- A találatokat nem csak listázni tudjuk, hanem valamilyen parancsot is végre tudunk rajtuk hajtani.
- find [egyéb paraméterek] -exec parancs ”{}” \;
  - A parancs fog majd lefutni.
  - A ”{}” helyére helyettesítődik be a talált fájl/könyvtár.
  - A \; arra való, hogy a -exec paraméterlistáját lezárja.
- A -exec után a legtöbb tanult parancs valami értelmeset fog csinálni.

Keresés (find) – VI.

- Példák a find -exec-re
  - -exec ls -l ”{}” \; Minden megtalált elem listázása hosszan.
  - -exec ls -ld ”{}” \; Ugyanaz, de a könyvtáraknál nem a tartalom látszik.
  - -exec head -n 5 ”{}” \; Minden megtalált fájl első 5 sorának kiírása.
  - -exec tail -n 8 ”{}” \; Minden megtalált fájl utolsó 8 sorának kiírása.
  - -exec cat ”{}” \;
    Minden megtalált fájl tartalmának kiírása
  - -exec wc -l ”{}” \; Minden megtalált fájl sorainak megszámolása. Ki fogja írni a fájlneveket is. (A -w, -c kapcsolókkal is használható.)

Keresés (find) – VII.

- Példák a find -exec-re
  - -exec grep minta ”{}” \; A mintára illeszkedő sorokat írja ki minden fájlból
  - -exec rm ”{}” \; A megtalált fájlok törlése
  - -exec cp ”{}” konyvtar \; A megtalált fájlok másolása a konyvtar könyvtárba.
  - -exec sort ”{}” \; A megtalált fájlok tartalmát egyenként rendezve kiírja.
  - -exec mv ”{}” ”{}”.old \; A megtalált fájlokat átnevezi. (A végére teszi hogy .old.) A {} jel többször is használható.
  - -exec chmod a+x ”{}” \; Minden fájlra futási jogot ad.

Keresés (find) – VIII.

- A find után használhatunk pipe-ot is, például ha a találatokat meg akarjuk számolni, vagy rendezni...
  - find ... | wc -l Megszámolja, hogy hány találat van.
  - find ... -exec cat ”{}” \; | wc -l
    Megszámolja, hogy összesen hány sor van az illeszkedő fájlokban.
  - find ... -exec cat ”{}” \; | sort Az illeszkedő sorokat rendezi.
  - find ... -exec cat ”{}” \; | grep minta | sort
    Az illeszkedő sorokban megkeres egy mintát, majd rendezi a sorokat.
  - find ... -exec ls -l ”{}” \; | awk '{print $9 ”\t” $5}' | sort -k 2 -n
    A leghosszabb fájlt keresi meg (rekurzívan).
  - find ... -exec head -n 1 ”{}” \; | sort A megtalált fájlok első sorait rendezi.
  - find ... -exec wc -l ”{}” \; | sort -n A megtalált fájlokat aszerint rendezi, hogy hány sorból állnak.

Keresés (find) – IX.

- A find a -exec utáni programot minden találatra lefuttatja, ami lassú lehet. A xargs paranccsal ez gyorsítható.
- Az xargs a standard bemenetről vett adatokból a megadott program számára paraméterlistát készít.
- Minden megtalált fájl törlése
  - find … -exec rm ”{}” \;
    Az rm minden alkalommal elindul.
  - find … | xargs rm Az rm egyszer fog futni.
  - find … -print0 | xargs -0 rm
    Minden fájlnevet biztosan helyesen fog kezelni.

Keresés (find) – X.

- Pipe és -exec
  - find … -exec head -n 3 ”{}” | tail -n 1 \;
    A fájlok 3. sorát írná ki, de ez így nem működik.
  - find … -exec sh -c 'head -n 3 ”{}” | tail -n 1' \;
    Indítunk egy belső shellt (sh), aminek megadjuk, hogy milyen parancsot futtasson (-c). Így pipe-os parancsot is tudunk futtatni az exec paramétereként.

Keresés (find) – XI.

- Keresés tartalom alapján
  - find . -name "*.conf" -exec sh -c 'if grep -q minta "{}"; then echo "{}"; fi' \;
  - A grep -q nem írja ki az illeszkedő sort, csak a visszatérési értékét állítja be aszerint, hogy volt-e találat, vagy nem.
  - Az if majd egy későbbi anyagban lesz részletesen tárgyalva.