5. óra

## Változók

Környezeti, pl.: PATH (mappák listája, ahol a parancsokat megtalálhatja a shell), env (összes környezeti változó)

Shell változók csak a shell-ben láthatóak.

Tartalma csak karaktersorozat lehet. (25 helyett "25")

Scriptek egymás változóit nem láthatják, mert minden script saját shellben indul. (Kivéve, ha környezetesítjük)

Neve betűvel kezdődik, majd jöhet "\_" és számok. (".", "-", stb. nem lehet.)

Változó tartalmára a $ jellel lehet hivatkozni.

Környezeti változóvá alakítás: export.

alma=Hello World  
echo $alma  
export alma

## Escaping, idézőjelek

\ megszünteti a következő karakter hatását.

'egyszeres idézőjelen belül minden hatás megszűnik'

"kétszeres idézőjelen belül a $, a \, és a ' karakterek hatása megmarad"

echo $alma  
Hello World

echo \$alma  
$alma

$almaalma  
nincs ilyen változó (helyköz kell az $alma után)

${alma}  
Hello World

unset alma  
kitörli

x=${alma-Hello World}  
ha nem létezik alma, létrehozza a megadott alapértelmezett értékkel

x=${alma=Hello World}  
még ha létezik is alma, megváltoztatja a megadott értékre

x=${alma?Hello}  
ha nincs $alma, kiírja, hogy Hello

## Parancsbehelyettesítés

`parancs` vagy újabb shelleken $(parancs)

ki\_vagyok=`whoami`

a=`date`; b='date'  
csak az a változat jó, a b string, "date" lesz.

eval b  
így lefut a date

## Fájlnevek

nem ékezetes betűk, számok, "\_", "-", "."

## Regexp-szerű speciális karakterek fájlnevekhez

? egy karakter

\* bármi, akár nullaszor is

[abc] felsoroltak közül egyet

[!abc] felsoroltak nem szerepelhetnek

[A-Z] vagy [1-9] intervallum

## Első shell scriptem (már akinek…)

vi elso (Beleírom, hogy "echo Ez az első scriptem!")  
chmod +x elso (futtatható legyen)

./elso  
Ez az első scriptem!  
elso (A shell csak a PATH könyvtárból keres parancsokat, aktuális könyvtárból nem. Érdemes a PATH módosítását elvégezni: pontot adni hozzá, hogy a jelenlegi mappát is magában foglalja: "PATH=$PATH:." a .profile állományba, hogy mindig lefusson.)

Elejére kerülhet még a: "#!/bin/sh" Ez meghatározza a végrehajtó állomány nevét. Amit ide beírunk, az fogja lefuttatni a scriptet.  
Majd új sorba egy üres komment "#", mert a mai shellekben előírás.

## Kimenet, bemenet

stdin (0) billentyűzet, alapértelmezett bemenet  
stdout (1) monitor, alapértelmezett kimenet  
stderr (2) monitor, alapértelmezett hibakimenet

## Átirányítás

Kimenet ">" jel segítségével.  
Pl.: echo alma barack szilva > gyümölcsök  
ehco alma &2 (hibakimenet)

Bemenet "<" jel segítségével  
Pl.: mail juli < level.txt

## Hozzáfűzés

Két darab << vagy >> jel segítségével.  
echo dió >>gyümölcsök (így nem írja felül, csak hozzáadja. "append")

echo <<alma  
<html><head><title>…  
alma  
(almától almáig minden, speciális karaktereket figyelmen kívül hagyva. HTML kód beírását nagyon megkönnyíti.)

## Hibakimenet átirányítása

cp 2>err.log (fájlt létrehozza, beleírja a hibát: nem adtuk meg mit másoljon)  
valami 2>>err.log (fájlhoz hozzáfűzi a hibát: nincs ilyen parancs)

## Pipe (csövek)

| Bemenetet olvas, kimenetre ír.

Fontosabb szűrők: cut tee, sort, wc, grep

Ideiglenes eredményfájl. Ez is jó, de kezdő megoldás  
who >nevek #A nevek fájlba írja a who parancs eredményét.  
sort nevek #Sorba rendezi a neveket.

Profik összekapcsolják csövekkel (pipe-olás)  
who|sort

További eszközök:  
who|sort -r -u #-r fordított sorrend, -u egyedi sorok (nem lehet két sor egyforma)  
who|wc -l #megszámolja

## Parancsok felhasználói kézikönyve: man(ual)

man sort  
man -k kulcsszo (keresés)

## Cut

Inputból vagy fájlból mezőket, oszlopokat vághatunk ki.

cut -c1-5 (1-5 karakteroszlop kivágása)  
Pl.: date|cut -c4-8 (eredménye Oct)

cut -f1,3,5-7 (1,3,5,6,7 mezők kivágása)

Alapértelmezett mező a " " (tab)

Új mezőelválasztót a "-d char" kapcsolóval tudunk megadni.

Pl.: cat /etc/passwd|cut -f1,7 -d: (név és shell oszlopokat adja vissza))

## Grep (a szűrők szűrője)

Megadott mintával rendelkező **sorok** kiválasztása. (grep Hello World)

Fontos paraméterei:  
-v Mintát nem tartalmazó sorok.  
-i Kis és nagybetűket nem különböztet meg (ez az alapértelmezett egyébként).  
-w Csak önálló szóként találja meg (pl grep -w Pista nem találja meg a traPista elemet).  
-r Rekurzívan keres az egész könyvtárban, amit megadtunk.  
-l Csak fájlneveket ad vissza (fájlban keres).  
-c Csak a sorok számát írja ki.  
-n Megszámozza a sorokat.

cat névsor|grep Pista (minden Pistát tartalmazó sor)

grep -r 'fradi' ./script (script könyvtárban a fradis sorokat keresi)

## Regexp használható a grep-ben

^ Sor elejétől kell egyezni a mintának.

$ Sor végétől kell egyezni a mintának.

. (Pont) egy tetszőleges karakter.

\* Előző minta ismétlése (0-szor vagy többször). A minta lehet [kapcsos zárójelek] között is. Vagy akár pont. (Más helyeken a "\*" a wildcard, itt a ".\*" a wildcard.)  
Pl.: ^a.\*fa$ (minden sor, ami a-val kezdődik és fa-ra végződik. Pl.: afa, almafa, az egy öreg fa)

[a-zA-Z0-9] alfanumerikus (röviden "\w") (És a "\W" nem alfanumerikus karaktert jelent.)

\d számjegy (digit)

\s szóköz, tab, sortörés (white space)

Szóillesztés:  
\< szókezdet  
\> szóvég

## Bővített grep

egrep (expendív, grep -E) fgrep (fix, grep -F)

Vagy: alma|fa

Legalább egyszeri ismétlődés: + "p+f" (legalább egy "p", majd egy f: pf ppf ppf)

Nullaszor vagy egyszeri ismétlés: ?

{5} pontosan ötször ismétlődjön az előző minta.

[a-d]\w{5} a-d-vel kezdődő, majd 5 alfanumerikus karakter jön.

{2,4} 2 3 vagy 4-szer ismétlődjön

Minta csoportba foglalása: () Pl.: bankszámla: [0-9]{8}(\s[0-9]{8}){1,2} 8 szám, majd még egyszer vagy kétszer helyköz és 8 szám. ([0-9] helyett \d is jó.

Speciális karakterek elé \-t kel rakni. Pl pont, hogy ne wildcard legyen így kell: "\."

További lehetőségek a man(ual)-ban vannak.