07

BASH script programozás III.
haladó eszközök (gyakorlás), szűrők, linkelés

zh (március 27.)

- felépítése
  - három feladat (6-6-8)
- anyagrész: minden, ami a félév során elhangzott.
  - BASH scripteket kell írni,
  - ismerni kell a vezérlési szerkezeteit, paraméter kezelést, változó használatot, matematikai jelölést, parancs kimenetre hivatkozást
  - ismerni kell az összes, gyakorlaton megnézett parancsot, például grep, cut, wc, chmod, ...
  - ismerni kell az átirányítást, a csővezetéket, a felhasználókezelést és az alap mintaillesztést
- amit használhatsz (ne bízd el magad, a zh-n való lapozgatása nem lesz elég, ahhoz hogy megold a feladatokat):
  - Rodek Lajos gyakorlati jegyzete

Környezeti változók

- környezet:
  - név – érték párok halmaza
- (globális) shell szintű változók
- env, printenv: környezeti változók listázása
- export VALTOZO_NEV vagy export VALTOZO_NEV=érték:
  - környezeti változó létrehozása
- unset VALTOZO_NEV: környezeti változó feloldása

Logikai műveletek

```
case $VÁLTOZÓ in
    minta1)
        PARANCS(OK);;
    minta2|minta3)
        PARANCS(OK);;
    *)
        PARANCS(OK)
esac
```

while, until

while [FELTÉTEL **]; do**

PARANCS(OK)

done

- vagy másképpen

while ((FELTÉTEL **)); do**

- PARANCS(OK)

done

- ehhez hasonlóan létezik **until** vezérlési szerkezet (amíg nem teljesül a feltétel)

függvények, függvényhívás

```
function FUGGVENYNEV() {
    PARANCS(OK)
}
```

FUGGVENYNEV PARAM1 PARAM2 … PARAMN

if - kiegészítés

- if [kapcsolós feltétel]; then √
- if ((matematikai kifejezéses feltétel)); then √
- if PARANCS; then
  - a PARANCS kilépési állapota alapján lesz igaz vagy hamis →
    - hiba nélküli futás: true
    - hiba esetén pedig: false

Kilépési állapot (exit state)

- Általában a kilépési állapot:
  - exit 0 ha minden OK (if esetén true)
  - exit 1 valamilyen hiba történt (if esetén false)
  - exit 2-255 egyéb hiba
- Az exit status 0 – 255 vehet fel értéket, tehát van lehetőség saját hiba kilépési státusz használatára is (kivételkezelés)
- PARANCS

  if (($? == 0)); then

  #Ahol a $? az előző PARANCS exit állapotára hivatkozó változó.

Feladat

- hozzunk létre könyvtárat (név fogadása paraméterről)
  - ha sikeres, akkor írjuk ki, hogy SUCCESS
  - ha már létezik a fájl, vagy bármi hiba történik, akkor pedig azt, hogy ERROR

    (a többi hibakimenetre írt tartalmat nyeljük el /dev/null)

Szűrő

- grep – csak a mintához illeszkedő sorokat írja fájlból
- kapcsolók:
  - n: kiírja az illeszkedő sor sorszámát
  - c: megszámlálja azokat a sorokat, ahol illeszkedést talált

Feladat:

- /etc/mime.types -ban keressük meg a video kiterjesztéseket, hány ilyen sor van?
- töltsük le a gyakorlat honlapjának a html kódját és keressük meg benne a „zh” kifejezést.

cut

- sorok kiválasztott részeit írja ki. Kapcsolók:
  - -b megadott bájtok közötti részt írja ki
  - -c megadott karakterszám közötti részt írja ki
  - **-d elválasztó jel (alapértelmezett a TAB)**

    -f elválasztással született megadott mezők közötti részt írja ki
- Feladat:
  - cutoljuk az ls -l tartalmát (pl első karakter)
  - cutoljuk a ls -1 tartalmából az állományneveket, kiterjesztéseket

Linkelés

- egy állományt több helyen/néven is elérhetővé tegyünk a fájlrendszerben
- Hard link:
  - ln TARGET LINKNAME
  - újra létrehozza a fájlt, ugyanarra az inodera mutat, bármelyik változik akkor egyszerre változik az összes
  - **biztonsági másolat (tükrözés)**
  - eredeti törlése esetén megmarad a hard link
  - könyvtárra nem lehetséges
- Soft (v. symbolic) link:
  - ln -s TARGET LINKNAME
  - hagyományos értelemben egyszerű link (tükör) jön létre, ami az eredeti állományra mutat
  - **hagyományos link**
  - eredeti törlése esetén megmarad a soft link, de sérül és nem működik többet

Feladat: egy tetszőleges állományról hozzunk létre mindkétféle linket és töröljük az eredetit, vizsgáljuk meg a különböző linkeket