

06

BASH script programozás II.
Vezérlési szerkezetek

Emlékeztető

- Jelölésbeli különbség van parancs végrehajtása és a parancs kimenetére való **hivatkozás** között

PARANCS ↔ **\$(PARANCS)**

- Jelölésbeli különbség van változó értékadás és változó értékére való **hivatkozás** között

VARIABLE=10 ↔ **\$VARIABLE** (vagy **\${VARIABLE}**)

- Jelölésbeli különbség van matematikai számítás elvégzése és ugyanennek az eredményére való **hivatkozás** között

((3+2)) ↔ **\$((3+2))**

if – Parancs végrehajtás

```
if PARANCS ; then
```

```
    PARANCS(OK)
```

```
elif PARANCS ; then
```

```
    PARANCS(OK)
```

```
else
```

```
    PARANCS(OK)
```

```
fi
```

A parancs exit status alapján értékeli ki (\$?).

if - Logikai műveletek (1.)

if [FELTÉTEL] ; then

 PARANCS(OK)

elif [FELTÉTEL] ; then

 PARANCS(OK)

else

 PARANCS(OK)

fi

!! [] zárójel előtt, után **KELL a SZÓKÖZ !!**

!! FELTÉTEL megadása csak speciális **kapcsolókkal** lehetséges

!! Hagyományos módon (>) **átirányítás** fog történni !!

Feltételes kifejezések

- Numerikus összehasonlítás:
 - KIF1 -eq KIF2 (equal ~ egyenlő)
 - KIF1 -ne KIF2 (not equal ~ nem egyenlő)
 - KIF1 -lt KIF2 (less than ~ kisebb mint)
 - KIF1 -le KIF2 (less or equal than ~ kisebb egyenlő)
 - KIF1 -gt KIF2 (greater than ~ nagyobb mint)
 - KIF1 -ge KIF2 (greater of equal than ~ nagyobb egyenlő)
- Logikai kifejezések:
 - [KIF1 -a KIF2] (and)
 - [KIF1 -o KIF2] (or)
 - [! KIF1]

Feltételes kifejezések

- Szöveges összehasonlítás (a **KIF** helyére string/szöveg kerül)
 - KIF1 == KIF2
 - KIF1 != KIF2
 - -z KIF (üres szó)
 - -n KIF (nem üres szó)
- Állományok jellemzőinek a vizsgálata (a **KIF** helyére egy állomány neve kerül):
 - -e KIF (exists ~ létezik), -d KIF (directory ~ létezik és könyvtár), -f KIF (file ~ létezik és közönséges állomány), -L KIF (létezik és szimbolikus link)
 - -r, -w, -x KIF olvasási, írási és végrehajtási jog az aktuális felhasználó szemszögéből
 - -O, -G KIF tulajdonos, csoport megegyezik-e az aktuális felhasználóval
 - -s KIF (nem üres fájl)
 - KIF1 -nt KIF2 (newer than ~ újabb mint)
 - KIF1 -ot KIF2 (older than ~ régebbi mint)

if - Logikai műveletek (2.)

```
if (( FELTÉTEL )) ; then  
    PARANCS(OK)  
elif (( FELTÉTEL )) ; then  
    PARANCS(OK)  
else  
    PARANCS(OK)  
fi
```

Feladat

- Az első két paraméterként kapott szám közül írjuk ki a maximális értékűt (mindkétféle if jelöléssel próbáljuk ki).
- Az első két paraméterként kapott fájlról, döntsük el, hogy melyik az újabb. Majd az újabbról írjuk ki, hogy mappa, vagy közöséges fájl-e.

for (lista alapú)

- lista bejárása:

```
for VÁLTOZÓ in LISTA; do
```

```
    PARANCS(OK)
```

```
done
```

- LISTA megadása:
 - szöveg felsorolva szóközzel elválasztva
 - mintaillesztéssel fájlok listája
 - `$()` ← beágyazott parancs kimenete (ami szóközös vagy több soros kimenetű)
 - változóhivatkozás (olyan változó, ami szóközös vagy többsoros tartalommal rendelkezik)
- Feladat: próbáljuk ki az összeset

for (index alapú)

- index alapú for:

N=10

```
for (( i=0; i<=$N; i++ )); do
```

```
    PARANCS(OK)
```

```
done
```

- ritkábban fogjuk használni !

Tetszőleges ciklusnál használható: a **break**,
continue és az **exit**

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)

while – végtelen ciklus

```
while true ; do
```

```
    PARANCS(OK)
```

```
    if [ FELTÉTEL ] ; then
```

```
        break;
```

```
    fi
```

```
done
```

Feladatok

- A paraméterként megadott állományok közül számoljuk össze a közöséges állományokat
(az összes paraméter listája: \$*)
- Az eredményt egy fájlban tároljuk
- ++ Ugyanezt while ciklussal, de ne paraméterről várjuk az állományokat, hanem standard bemenetről olvassuk be őket, egészen addig amíg „q” vagy „quit” kulcsszót nem ír be a felhasználó