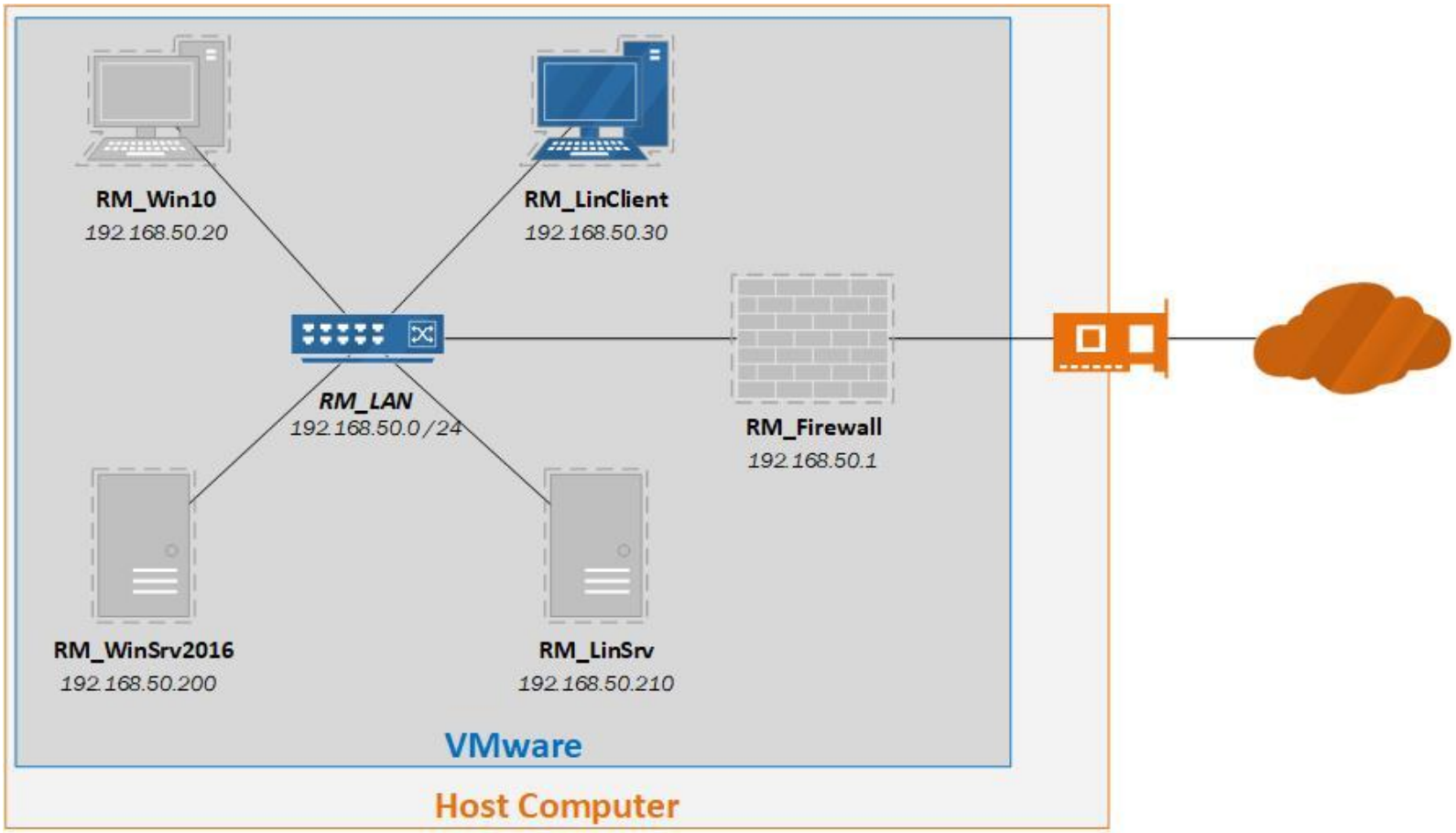
Magyarország Kormánya
Európai Unió
Európai Szociális Alap
BEFEKTETÉS A JÖVŐBE
SZÉCHENYI 2020

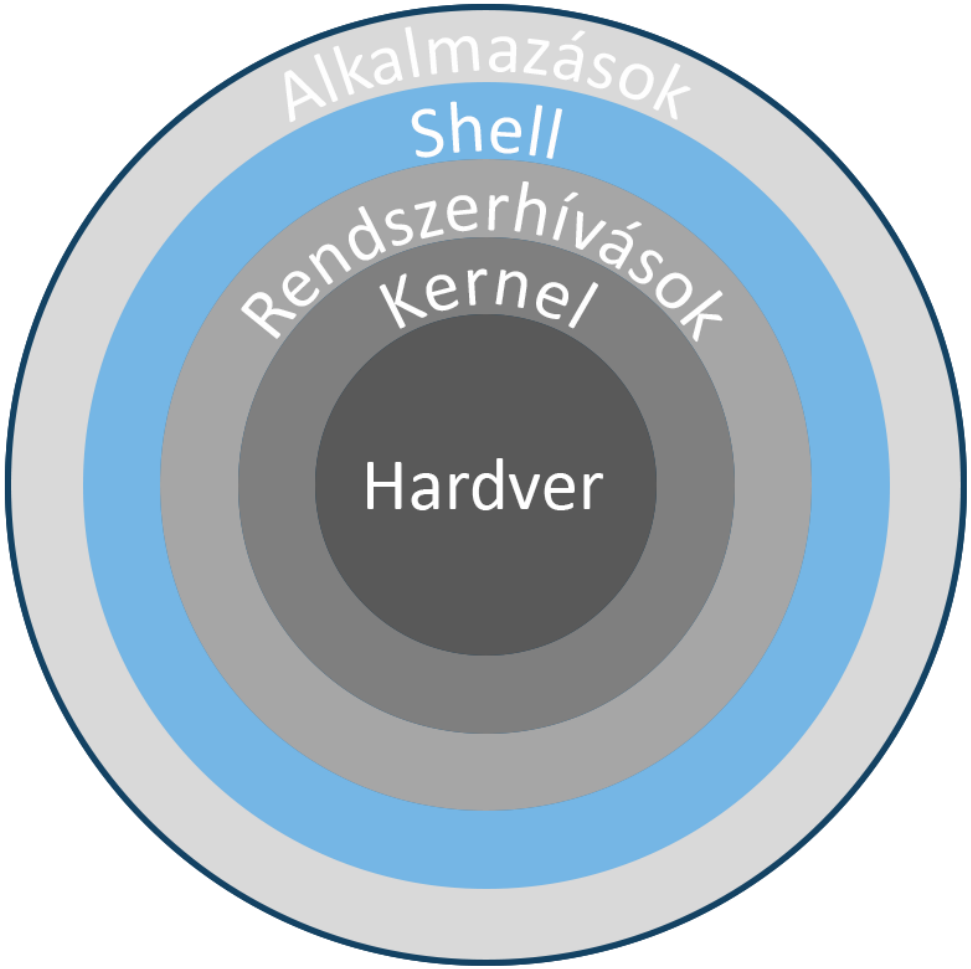
Linux script - alapok

Operációs rendszerek

Óbudai Egyetem

A szükséges virtuális gépek




Alkalmazások
Shell
Rendszerhívások
Kernel
Hardver

Mik azok a scriptek?

- Rövid programok, melyek parancssoros utasítások sorozatát tartalmazzák
- Ezeket egy szöveges fájlba írjuk
- Gyakran automatizálunk scripteket feladatok elvégzésére (pl. mentések, frissítések)

Első script létrehozása

Script fájl alapok – 1/2

- Scriptet bármilyen grafikus vagy parancssoros szövegszerkesztő programban létrehozhatunk
  - vi
  - nano
  - mcedit
  - Stb.
- A shell scriptek első sora az ún. shebang, ami megmondja, hogy melyik parancsértelmezőt használja az operációs rendszer a scripthez

```
#!/bin/sh
```

- A scriptbe kommenteket a # jel után tudunk tenni

```
# Ez egy komment
```

Script fájl alapok – 2/2

- A scriptek végén érdemes az exit utasítást használni (kilép a scriptből), mert ezzel tudjuk jelezni, hogy a script futása közben volt-e hiba (nem kötelező)

  `exit 0` – nem volt hiba a script lefutása közben

  `exit 1` – hiba volt a script lefutása közben

- A létrehozott scripten a felhasználónak futtatási joggal kell rendelkezzen

  `chmod a+x` `<fájlnév>`
  - minden felhasználónak ad jogot futtatni

- Futtatni a scriptet a `./<fájlnév>` -vel tudjuk

Első script

Feladat: Írjunk egy olyan scriptet, amely kiírja a „Hello World!”-öt

- Az **echo** parancsot használjuk szöveg kiírására
- Nyissunk meg egy szövegszerkesztőt
- A script a következő sorokból áll:

```
#!/bin/sh

echo ”Hello World!”
exit 0
```

- Mentsük el a felhasználó saját mappájába **hello.sh** néven
- Adjunk neki futtatási jogot (**chmod a+x hello.sh**)
- Futtassuk a scriptet

```
student@linclient:~$ ./hello.sh
Hello World!
student@linclient:~$
```

Dátum kiírása

- A scriptek több parancsot szoktak tartalmazni, melyek egymást szekvenciálisan követik

Feladat: Írjunk egy olyan scriptet, amely kiírja a „Az aktuális dátum:” szöveget és alá kiírja a dátumot!

- Dátumot kiíratni a **`date`** paranccsal tudunk
- Mentsük el a felhasználó saját mappájába **datum.sh** néven

Dátum kiírása (megoldás)

- A script a következő sorokból áll:

```
#!/bin/sh

echo ”Az aktuális dátum:”
date
exit 0
```

- Futtatás után a következő az eredmény:

```
student@linclient:~$ ./datum.sh
Az aktuális dátum:
2018. aug. 23., csütörtök, 09:49:18 CEST
student@linclient:~$
```

Változók

Változók

- Scriptekben van lehetőségünk értékeket változókba menteni
- A változók nem típusosak, nincs definiálva, hogy szám vagy szöveg lesz benne
- Változót létrehozni a következőképpen tudunk:

```
szam=12

szo=”húsleves”

szoveg=”egy tál húsleves”
```

- Az = jel köré nem szabad space-t tenni!
- Szám értékeket közvetlenül, szöveget ” ” közé kell írni!
- Változó értékét később a $ segítségével tudjuk kiolvasni:

```
echo $szoveg
```

Változó használata 1.

Feladat: Írjuk át a hello.sh (Hello World-öt kiíró) scriptet úgy, hogy a kiírandó szöveget egy változóból szedje!

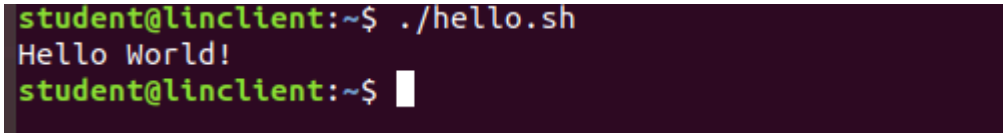
Változó használata 1. (megoldás)

- A script a következő sorokból áll:

```
#!/bin/sh

szoveg=”Hello World!”
echo $szoveg
exit 0
```

- Futtatás után a következő az eredmény:



Változó használata 2.

- Az echo parancs több string-et is ki tud írni egy sorban

```
echo ”Első string” ”Második string”
```

- Az eredmény nem egészen összefűzés, mert space lesz a két string között!

```
student@linclient:~$ echo "Első string" "Második string"
Első string Második string
student@linclient:~$
```

Feladat: Írjuk át a hello.sh (Hello World-öt kiíró) scriptet úgy, hogy külön változóban tároljuk a két szót (külön a Hello-t és külön a World-öt)!

Változó használata 2. (megoldás)

- A script a következő sorokból áll:

```
#!/bin/sh

szoveg1=”Hello”
szoveg2=”World!”
echo $szoveg1 $szoveg2
exit 0
```

- Futtatás után a következő az eredmény:

```
student@linclient:~$ ./hello.sh
Hello World!
student@linclient:~$
```

Beépített változók

- Vannak speciális, beépített változók:
- **`$?`** – az előző parancs vagy script lefutásának eredménye (exit)
  - 0 az értéke, ha sikeresen lefutott
- **`$#`** – a megadott paraméterek száma
- **`$1 , $2 , … , $9`** – az n.-edik paraméter értéke
- **`$0`** – az adott script neve
- **`$*`** – az összes paraméter egyben (egy stringként)
- **`$@`** – az összes paraméter egyben (egy tömbként)

- Emellett vannak környezeti változók, melyeket listázni a **`printenv`** paranccsal tudunk

Speciális változó használata 1.

Feladat: Írjuk át a hello.sh scriptet úgy, hogy a kiírandó két szót paraméterként adjuk meg! Csak beépített változót használjunk!

Speciális változó használata 1. (megoldás)

- A script a következő sorokból áll:

```
#!/bin/sh

echo $1 $2
exit 0
```

- Futtatás után a következő az eredmény:

```
student@linclient:~$ ./hello.sh Hello World!
Hello World!
student@linclient:~$
```

Speciális változó használata 2.

Feladat: Írjuk át a hello.sh scriptet úgy, hogy a paraméterként megadott szavak kiírása után írja ki a paraméterek számát!

Speciális változó használata 1. (megoldás)

- A script a következő sorokból áll:

```
#!/bin/sh

echo $1 $2
echo ”A paraméterek száma:” $#
exit 0
```

- Futtatás után a következő az eredmény:

```
student@linclient:~$ ./hello.sh Hello World!
Hello World!
A paraméterek száma: 2
student@linclient:~$
```

Érték beolvasása

- A felhasználótól futás közben is kérhetünk be értékeket
- Ehhez a `read` parancsot tudjuk használni

```
read bekertszoveg
```

  - A bekért érték a read után megadott változóban lesz eltárolva
  - A változót nem kell előre definiálni

Feladat: Módosítsuk a hello.sh scriptet úgy, kérje be a felhasználó nevét és név szerint köszönjön neki!

Érték beolvasása (megoldás)

- A script a következő sorokból áll:

```
#!/bin/sh

echo ”Mi az ön neve? ”
read nev
echo ”Hello” $nev
exit 0
```

- Futtatás után a következő az eredmény:

```
student@linclient:~$ ./hello.sh
Mi az ön neve?
Gergely Dániel
Hello Gergely Dániel
student@linclient:~$
```

Idézőjelek, parancs behelyettesítés

Idézőjelek, parancs behelyettesítés – 1/2

- A scriptekben három fajta idézőjelet használhatunk:
  - **’ ’** – az aposztrófok közötti szöveg string-ként van értelmezve és nem helyettesíti be a változók érétkét!

```
valtozo=”date”
echo ’$valtozo’
```

kimenet: **$valtozo**

  - **” ”** – a hagyományos idézőjelek közötti szöveg string-ként van értelmezve, de behelyettesíti a változók érétkét!

```
valtozo=”date”
echo ”$valtozo”
```

kimenet: **date**

Idézőjelek, parancs behelyettesítés – 2/2

- ` ` – az alt gr+7 típusú aposztróf közé helyezett szöveg behelyettesítéskor parancsként fut le

```
valtozo=”date”
echo `$valtozo`
```

kimenet: **2018. aug. 23., csütörtök, 10:46:19 CEST**

- A fenti példa esetén a két `` között a változóban megadott szöveg parancsként lefut
- Értékadáskor parancs kimenetét is át lehet adni

```
valtozo=`date`
echo $valtozo
```

a kimenet ugyan az lesz, de ebben az esetben a date parancs már a változó értékadásakor lefut!

- Értékadáskor parancs kimenetét másként is át lehet adni:

```
valtozo= date
echo $valtozo
```

Ebben az esetben kell a szóköz az = után, de az ``-os és a space-es verziókat ne vegyítsük!

Parancs behelyettesítés használata 1.

Feladat: Írjuk át a datum.sh scriptet úgy, hogy az „Az aktuális dátum:” szöveget és magát a dátumot egy sorban írja ki! Használjuk parancs behelyettesítést!

Parancs behelyettesítés használata 1. (megoldás)

- A script a következő sorokból áll:

```
#!/bin/sh

echo ”Az aktuális dátum:” `date`
exit 0
```

- Futtatás után a következő az eredmény:

```
student@linclient:~$ ./datum.sh
Az aktuális dátum: 2018. aug. 23., csütörtök, 11:01:53 CEST
student@linclient:~$
```

Parancs behelyettesítés használata 1. (egy másik megoldás)

- A script a következő sorokból áll:

```
#!/bin/sh

echo ”Az aktuális dátum: `date`”
exit 0
```

- Futtatás után a következő az eredmény:

```
student@linclient:~$ ./datum.sh
Az aktuális dátum: 2018. aug. 23., csütörtök, 11:01:53 CEST
student@linclient:~$
```

- A kimenet megegyezik, de a scriptből látható, hogy a különféle idézőjelek egymásba ágyazhatók!

Parancs behelyettesítés használata 2.

Feladat: Írjunk egy olyan scriptet, amely a mappa tartalmát kilistázza és annak kimenetét elmenti az aktuális dátumként elnevezett fájlba!

- A parancsok kimenetét a **>** operátor segítségével lehet fájlba menteni
  - A fájlt létrehozza, ha nem létezik
  - A létezik a fájl, akkor annak tartalmát felülírja
  - Ha nem szeretnénk a fájl tartalmát felülírni, hanem hozzá szeretnénk fűzni, akkor használjuk a **>>** operátort

```
echo ”Szöveg a fájlba” > fajl.txt
```

- Mentsük el a felhasználó saját mappájába **lista.sh** néven

Parancs behelyettesítés használata 2. (megoldás 1/2)

- A script a következő sorokból áll:

```
#!/bin/sh

ls -l > `date`.txt
exit 0
```

Parancs behelyettesítés használata 2. (megoldás 2/2)

- Futtatás után a következő az eredmény:

```
student@linclient:~$ ./lista.sh
student@linclient:~$ ls
'2018. aug. 23., csütörtök, 11:30:00 CEST.txt'   examples.desktop   Public
 datum.sh                                        hello.sh           Templates
 Desktop                                         lista.sh           Videos
 Documents                                       Music
 Downloads                                       Pictures
student@linclient:~$ cat 2018.\ aug.\ 23.\,\ csütörtök\,\ 11\:30\:00\ CEST.txt
total 56
-rw-r--r-- 1 student student    0 aug   23 11:30 2018. aug. 23., csütörtök, 11:
30:00 CEST.txt
-rwxr-xr-x 1 student student   53 aug   23 11:01 datum.sh
drwxr-xr-x 2 student student 4096 aug   11 15:59 Desktop
drwxr-xr-x 2 student student 4096 júl   16 16:06 Documents
drwxr-xr-x 2 student student 4096 júl   24 11:28 Downloads
-rw-r--r-- 1 student student 8980 júl   16 15:55 examples.desktop
-rwxr-xr-x 1 student student   62 aug   23 10:31 hello.sh
-rwxr-xr-x 1 student student   30 aug   23 11:29 lista.sh
drwxr-xr-x 2 student student 4096 júl   16 16:06 Music
drwxr-xr-x 2 student student 4096 aug   11 15:59 Pictures
drwxr-xr-x 2 student student 4096 júl   16 16:06 Public
drwxr-xr-x 2 student student 4096 júl   16 16:06 Templates
drwxr-xr-x 2 student student 4096 júl   16 16:06 Videos
student@linclient:~$
```

Parancs behelyettesítés használata 3.

- A kimeneti fájl neve nem néz ki szépen, de szerencsére a `date` parancs kimenete paraméterekkel formázható! A formázással kapcsolatban a `date --help` parancs segítségével további információkat kapunk.

Feladat: Módosítsuk a lista.sh scriptet úgy, hogy a kimeneti fájl a következőképpen nézzen ki: év-hónap-nap_óra-perc.txt!

Parancs behelyettesítés használata 3. (megoldás)

- A script a következő sorokból áll:

```
#!/bin/sh

ls -l > `date +%Y-%m-%d_%H-%M`.txt
exit 0
```

```
student@linclient:~$ ./lista.sh
student@linclient:~$ ls
2018-08-23_11-41.txt  Documents          hello.sh  Pictures   Videos
datum.sh              Downloads          lista.sh  Public
Desktop               examples.desktop   Music     Templates
student@linclient:~$ cat 2018-08-23_11-41.txt
total 56
-rw-r--r-- 1 student student    0 aug   23 11:41 2018-08-23_11-41.txt
-rwxr-xr-x 1 student student   53 aug   23 11:01 datum.sh
drwxr-xr-x 2 student student 4096 aug   11 15:59 Desktop
drwxr-xr-x 2 student student 4096 júl   16 16:06 Documents
drwxr-xr-x 2 student student 4096 júl   24 11:28 Downloads
-rw-r--r-- 1 student student 8980 júl   16 15:55 examples.desktop
-rwxr-xr-x 1 student student   62 aug   23 10:31 hello.sh
-rwxr-xr-x 1 student student   53 aug   23 11:40 lista.sh
drwxr-xr-x 2 student student 4096 júl   16 16:06 Music
drwxr-xr-x 2 student student 4096 aug   11 15:59 Pictures
drwxr-xr-x 2 student student 4096 júl   16 16:06 Public
drwxr-xr-x 2 student student 4096 júl   16 16:06 Templates
drwxr-xr-x 2 student student 4096 júl   16 16:06 Videos
student@linclient:~$
```

Összetett parancs futtatása változóból

- Előfordulhat, hogy a változóban tárolt parancs, amit a script során futtatni szeretnénk összetett

```
parancs=”ls -l /home | sort”
```

- Ekkor nem tudjuk futtatni a ``-ok között

~~`$parancs`~~ ← hibára fut!

- Helyette használhatjuk az **eval** parancsot, ami a paramétereiből egyetlen parancsot rak össze és futtatja azt

eval $parancs ← nem fut hibára!

Paraméterként kapott parancsok futtatása parancsként

Feladat: Írjunk egy olyan scriptet, amely a bemenetként kapott paramétereket lefuttatja parancsként!

- Mentsük el a felhasználó saját mappájába **parancs.sh** néven

Paraméterként kapott parancsok futtatása parancsként (nem jó megoldás!)

- A script a következő sorokból áll:

```
#!/bin/sh

`$*`
```

- Ez egy nem jó megoldás!
- Példa: nézzük meg a következőt: ./parancs.sh ls
- Azt várnánk, hogy kilistázza a mappa tartalmát, de ami valójában történik az a következő:
  - Behelyettesül az ls, mint parancs, ami visszaadja a mappa tartalmát
  - A mappa tartalmát parancsként próbálja feldolgozni

```
student@linclient:~$ ./parancs.sh ls
./parancs.sh: 3: ./parancs.sh: 2018-08-23_11-41.txt: not found
student@linclient:~$
```

Paraméterként kapott parancsok futtatása parancsként (jó megoldás!)

- A script a következő sorokból áll:

```
#!/bin/sh

eval $*
```

- Futtatás után a következő az eredmény:

```
student@linclient:~$ ./parancs.sh ls
2018-08-23_11-41.txt  Documents         hello.sh  parancs.sh  Templates
datum.sh              Downloads         lista.sh  Pictures    Videos
Desktop               examples.desktop  Music     Public
student@linclient:~$
```

Matematikai műveletek

Matematikai műveletek

- A változóknak nincsenek típusai, minden string-ként van tárolva:

```
eredmeny=3+4
echo $eredmeny
```

kimenet: **3+4**

- A matematikai műveletek elvégzésére az **expr** parancsot használhatjuk:
  - Alap négy művelet egész számokon (+,-,*,/,%)
  - Logikai vizsgálatok (|,&)
  - Kisebb-nagyobb, egyenlőség, nem egyenlőség vizsgálata (<,<=,=,==,!=,>=,>)

```
szam1=3
szam2=4
eredmeny=`expr $sz1 + $sz2`
```

  - Fontos, hogy az operátor és az operandusok között space van!

Számok összeadása

Feladat: Írjunk egy olyan scriptet, amely a bemenetként két számot kap paraméternek és összeadja azokat!

- Mentsük el a felhasználó saját mappájába **szamologep.sh** néven

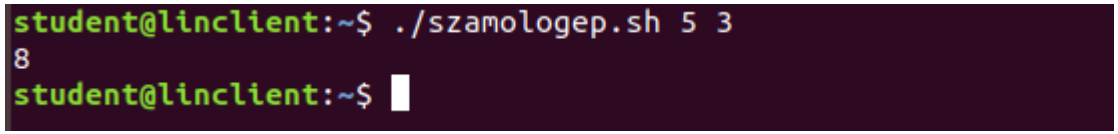
Számok összeadása (megoldás)

- A script a következő sorokból áll:

```
#!/bin/sh

expr $1 + $2
exit 0
```

- Futtatás után a következő az eredmény:



Számok szorzása

- **Feladat**: Módosítsuk a szamologep.sh scriptet úgy, hogy a paraméterként kapott két számot szorozza össze!

Számok szorzása (nem jó megoldás!)

- A script a következő sorokból áll:

```
#!/bin/sh

expr $1 * $2
exit 0
```

- Ez egy nem jó megoldás!
- A problémát a ***** okozza: helyére az aktuális mappa fájlneveit helyettesíti be, mi pedig ***** karaktert szeretnénk átadni
- Ahhoz, hogy az ilyen speciális karaktereket ”semlegesítsük” tegyünk eléjük egy **** karaktert
- Ilyen karakter még a **<** és a **>** is!

Számok szorzása (jó megoldás!)

- A script a következő sorokból áll:

```
#!/bin/sh

expr $1 \* $2
exit 0
```

- Futtatás után a következő az eredmény:

```
student@linclient:~$ ./szamologep.sh 5 3
15
student@linclient:~$
```

Köszönöm a figyelmet!

Operációs rendszerek

Óbudai Egyetem