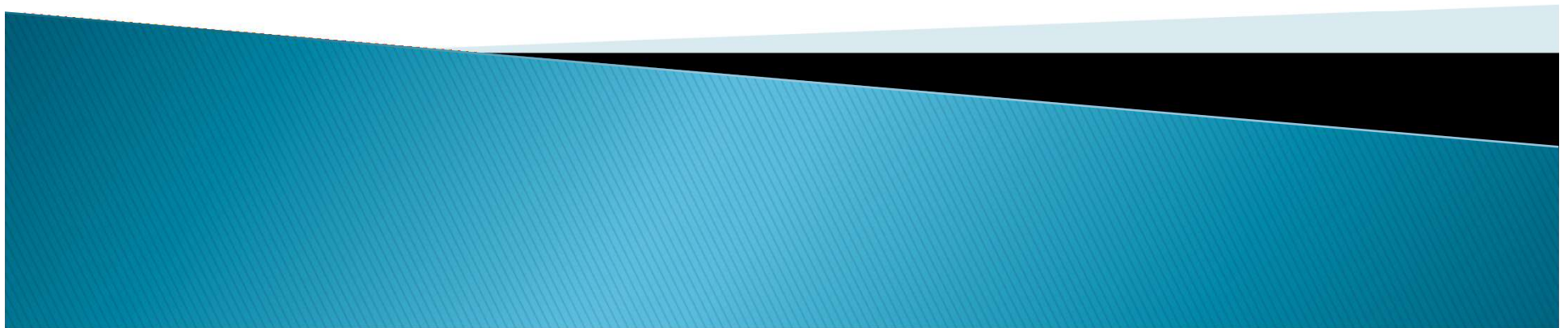
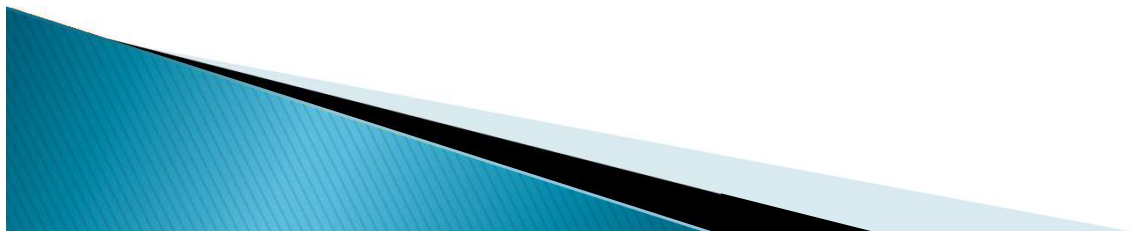


ZOS 5, 2017



Administrativa

- ▶ Toto a šesté cvičení – zkoušení skriptů
- ▶ Rozdělení na test – na šestém cvičení
- ▶ Sedmé cvičení – 1. test
- ▶ Courseware ZOS – Testy
 - Zde jsou uvedeny všechny podmínky



Úkol – skript case50.sh

Vytvořte skript, s využitím příkazu case, který:

- ▶ **-a** vypíše na obrazovku Dnes je: a dnešní datum (datum zjistí příkazem)
- ▶ **-b** uloží do souboru autor.txt login přihlášeného uživatele
- ▶ **-c** vypíše na obrazovku kolik měl skript parametrů
- ▶ **-d** vypíše na obrazovku počet řádek našeho skriptu



Zadání: skript kup.sh

- ▶ použití:
 - `./kup.sh` jahody rajce chleba "rohlik syrový" maslo
- ▶ činnost:
 - skript vytvoří soubor nakup.txt obsahující:

Chci nakoupit:

chleba
jahody
maslo
rajce
rohlik syrový

promyslete, jak zařídit, aby byli jednotlivé nákupní položky seříděné podle abecedy

- Přes pomocný soubor
- Bez pomocného souboru

Zadání: skript51.sh

Napište skript51.sh , který:

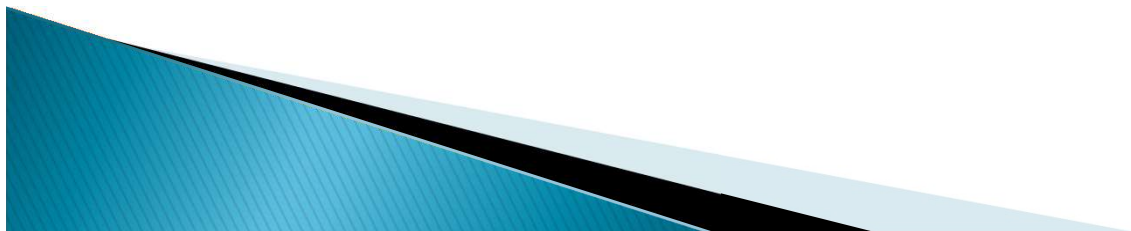
1. Vytvoří vždy soubor autor.txt obsahující login autora skriptu převedený na velká písmena
2. Při spuštění s `-a f` vytvoří symbolický link slink na soubor f
3. Při spuštění s `-b f` vypíše právě třetí řádku z f na obrazovku
4. Při spuštění s `-c f` nastaví práva na soubor f, že vlastník může vše, skupina jen číst, a ostatní nic
5. Při spuštění s jinými parametry vypíše text *Neplatný parametr!*



Zadání: skript52.sh

- ▶ Skript má dva parametry
 - 1. parametr: adresář
 - 2. parametr: řetězec
- ▶ Skript bude procházet zadaný adresář (1.par) a vypíše všechny názvy souborů, které uvnitř souboru obsahují zadaný řetězec (2.par)

`./skript52.sh . poklad`



Řešení: skript52.sh

```
#!/bin/bash
if test "$#" -ne 2
then
    echo "Zadej $0 adresar retezec"
    exit 1
fi
for A in "$1"/*
do
    if test -f "$A" ; then
        if test -r "$A" ; then
            cat "$A" | grep "$2" > /dev/null
            if test $? -eq 0 ; then echo "$A" ; fi
        fi
    fi
done
```

využijeme testu návratové
hodnoty, zda předchozí
příkaz skončil úspěšně

wget – stahování souborů z webu

- ▶ wget <http://www.zcu.cz>
 - Stáhne a uloží soubor index.html
- ▶ wget <http://nekde.cz/obrazek.jpg>
 - Stáhne soubor obrazek.jpg



Využití cyklu for pro stahování obrázků

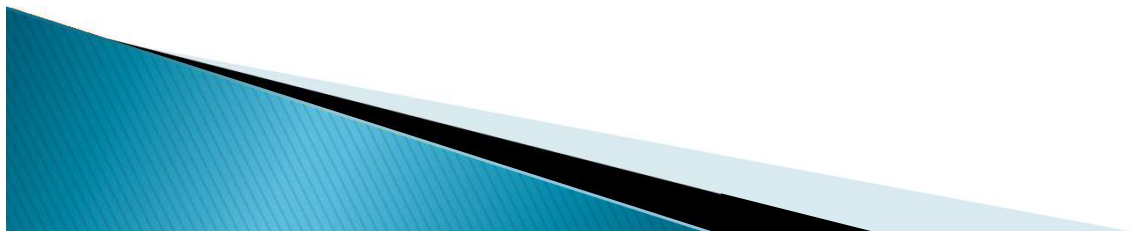
```
for a in $( seq 9 )  
do  
wget http://www.neco.cz/obr${a}.jpg  
done
```

Stáhne obr1.jpg až obr9.jpg

Příkaz seq x generuje čísla 1 .. x

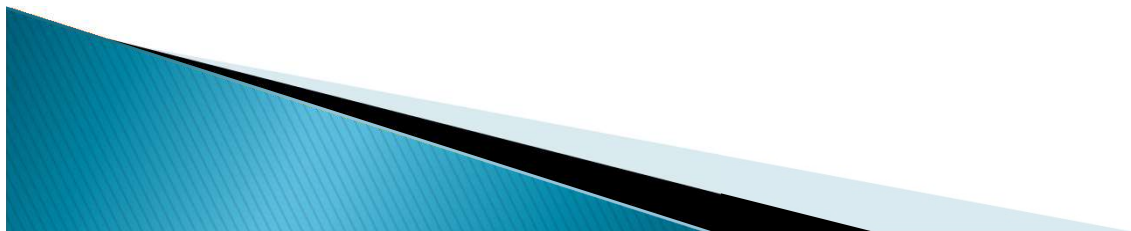
Znakové www prohlížeče, mail

- ▶ links <http://m.idnes.cz>
- ▶ lynx <http://m.idnes.cz>
- ▶ pine – práce s poštou např. na eryxu
- ▶ mail pesicka@kiv.zcu.cz –s pozdrav
 - Ahoj, jak se mas? <Ctrl>+<D>
- ▶ mail pesicka@kiv.zcu.cz < povidka.txt



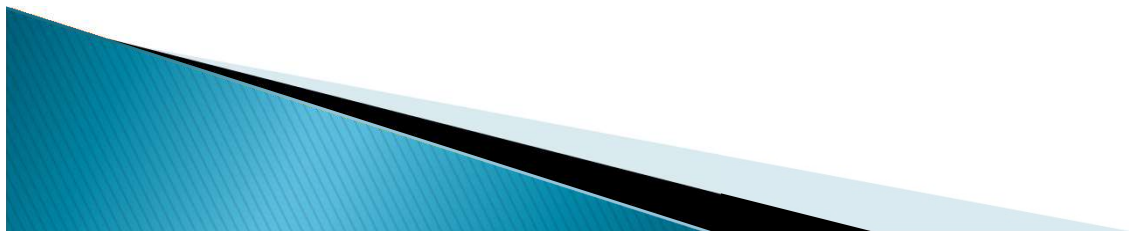
Spuštění procesu na pozadí

- ▶ `./vypocet &`
 - Spustí program výpočet na pozadí
 - V shellu můžeme zadávat další příkazy, nečeká na dokončení
- ▶ `nohup ./vypocet > vysledek.txt &`
 - Výpočet probíhá i po odpojení terminálu (např. zavřeme okno programu putty na eryx.zcu.cz)



Proměnné, uvozovky, apostrofy, zpětné apostrofy (!)

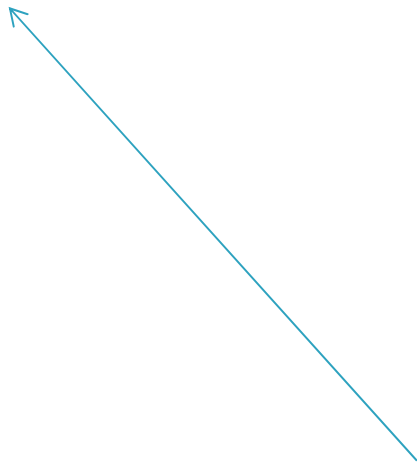
1. MOJE="Ahoj"
Pozor kolem znaku = nejsou mezery! Častá chyba
2. echo MOJE .. Vypíše MOJE
3. echo \$MOJE .. Vypíše Ahoj
4. touch \$MOJE x touch MOJE
5. echo "\$MOJE" .. vypíše Ahoj
6. echo '\$MOJE' .. apostrof, vypíše \$MOJE (!)
7. echo `ls` .. Zpětný apostrof, vykoná příkaz a nahradí jeho výstupem



Proměnné 2

- ▶ BARVA=zelenou
- ▶ echo "Mam rad \$BARVA barvu"
- ▶ echo 'Mam rad \$BARVA barvu'

- ▶ LIDI=\$(who)
- ▶ echo "prihlaseni: \$LIDI"
- ▶ echo prihlaseni: ` who `



Všimněte si rozdílu v interpretaci, pokud je řetězec mezi uvozovkami a apostrofem

Proměnné 3 – nezkoušet!

Pozor na nebezpečný příkaz !

Otestovat nanejvýš pouze v podadresáři!

- echo “ na tento prikaz pozor rm -rf * ”
 - Jen výpis
- echo “ na tento prikaz pozor ` rm -rf * ` “
 - **Nezkoušet, smaže!!**

Klasická ukázka code injection, kdy zdánlivě bezpečný příkaz echo může vést ke škodlivé činnosti

Co když zapomenou syntaxi for, case atd. ?

- ▶ Pokud máme jako aktuální shell bash (není-li tomu tak lze zajistit spuštěním */bin/bash*), můžeme získat nápovědu pro syntaxi interních příkazů takto:
 - ▶ `help for ; help if ; help case ; help echo ...`
 - ▶ `help test` ... jaké podmínky lze testovat
 - ▶ `man test`
 - ▶ `man bash` ... rozsáhlejší popis



Funkce v bashi – fce1.sh

```
#!/bin/bash
```

```
# prevzato z http://www.linuxexpres.cz/praxe/bash-23-dil
```

```
tento_pocitac()
```

```
{  
    echo -n "Pocitac: $HOSTNAME, cas: "  
    date  
}
```

```
echo "Obsazenost disku:"
```

```
tento_pocitac
```

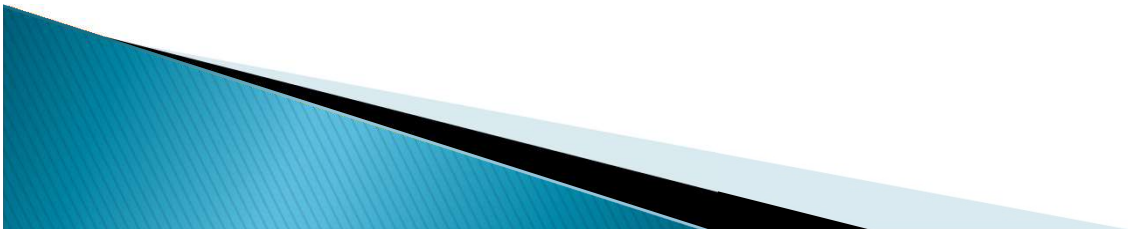
```
df -h
```

```
echo
```

```
echo "Prihlaseni uzivatele:"
```

```
tento_pocitac
```

```
who
```



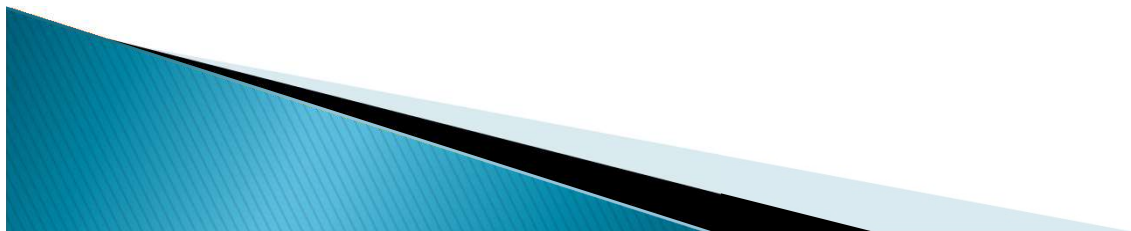
Funkce v bashi – návratová hodnota – fce2.sh

```
#!/bin/bash
```

```
# převzato z http://www.abclinuxu.cz/clanky/navody/bash-iv
```

```
vrat_retezec() {  
    echo "Řetězec"  
}
```

```
promena=$(vrat_retezec)  
echo $promena  
exit 0
```



Funkce – parametry – fce3.sh

- ▶ Při volání funkce poziční parametry např. \$1 nahrazeny parametry funkce

```
#!/bin/bash
```

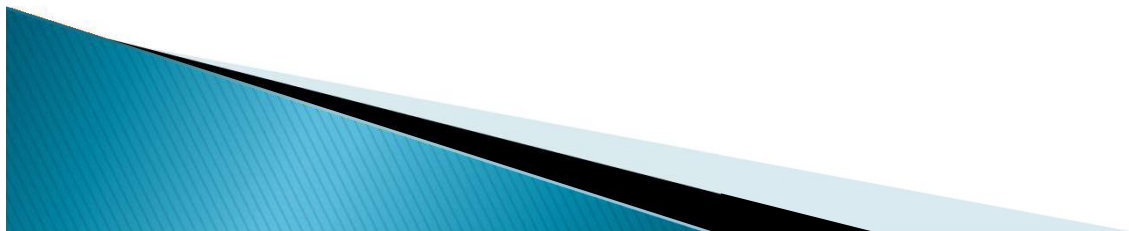
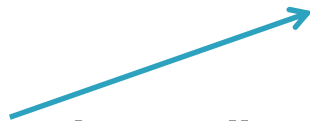
```
obed() {
```

```
    echo "Mam chut na $1"
```

```
}
```

```
obed "pecene kure"
```

```
obed rizek
```



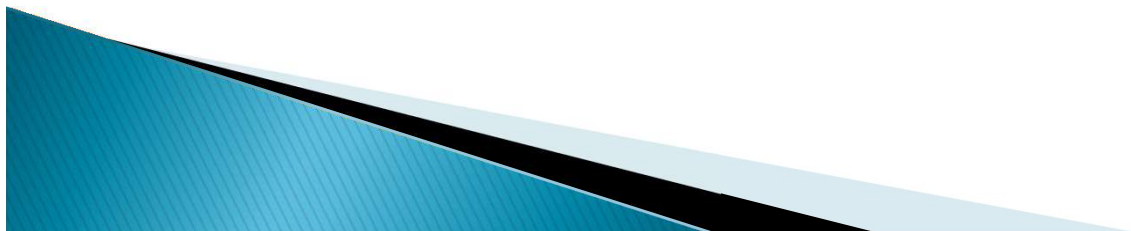
Iterace přes vstupní parametry

ite1.sh

```
#!/bin/bash
```

```
if [ $# -gt 3 ] ; then  
    echo "Vic nez 3 parametry"  
else  
    echo "Nanejvys 3 parametry"  
fi
```

```
for f in $@  
do  
    echo $f  
done
```



awk – manipulace s textem

- ▶ `who | awk '{ print $1 }'`
- ▶ `who | awk '{ print $5, $1 }'`

Máme text. soubor `kont.txt`
(jmeno prijmeni tel)

Chceme vypsát ve tvaru (prijmeni jmeno tel)

- ▶ `awk '{ print $2, $1, $3 }' kont.txt`
- ▶ `awk '{ print NR, $2, $1, $3 }' kont.txt`




Číslo řádky

awk

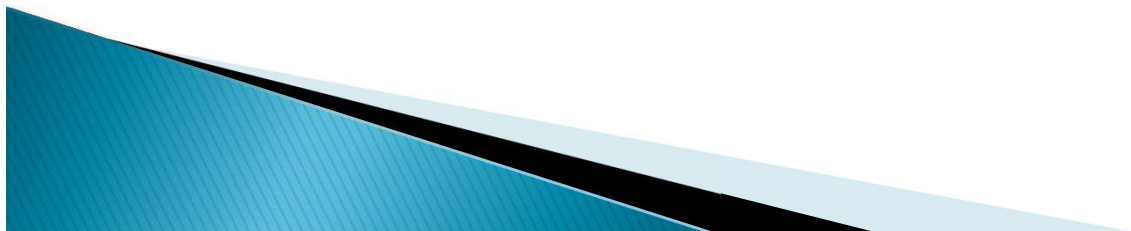
- ▶ `awk '/^Jan/ { print NR, $2,$1,$3}' kont.txt`
 - ▶ `awk '/Novak/ { print NR, $2,$1,$3}' kont.txt`

 - ▶ `/^Jan/` .. Vezme řádky začínající na Jan
 - ▶ `/Zluta$/` .. Řádka končí na Zluta

 - ▶ <http://cs.wikipedia.org/wiki/AWK>
 - ▶ <http://www.ucw.cz/~hubicka/skolicky/skolicka20.html>
- 

Napište skript `cv05.sh` v `bashi`, s následujícím chováním:

- ▶ Skript
 - vypíše Jsem spuštěný bez parametru (a skončí `exit 1`)
- ▶ Skript `-a`
 - vypíše na obrazovku Ahoj
- ▶ Skript `--pozdrav`
 - Vypíše na obrazovku Ahoj
- ▶ Skript `-u`
 - Uloží do souboru *lide.txt* seznam aktuálně přihlášených uživatelů
- ▶ Skript `-f f1 f2`
 - Uloží do souboru `f2` první a pátou řádku ze souboru `f1` (nápopěda – pro 5. řádku použijte např. kombinaci `head` a `tail`)



Pokračování zadání:

- ▶ Skript -v adr
 - Vypíše z adresáře pro každou položku, zda se jedná o soubor nebo adresář, tj.
soubor: ahoj.txt
adresar: adr1
- ▶ Skript -z f1
 - Spočte počet znaků v textovém souboru f1 a přidá do souboru znaky.txt následující záznam:
soubor: soub1.txt
znaku: 15
- ▶ Skript -s s1 s2 s3 s4
 - Spojí obsah textových souborů s1, s2, s3 do výsledného souboru s4
- ▶ Skript -t s1
 - Vypíše, zda je s1 obyčejný soubor, adresář, blokové nebo znakové zařízení

