ZOS 5, 2012

Zadání: skript kup.sh

- použití:
  - ./kup.sh jahody rajce chleba “rohlik syrovy“ maslo
- činnost:
  - skript vytvoří soubor nakup.txt obsahující:

```
Chci nakoupit:
------------
jahody
rajce
chleba
rohlik syrovy
maslo
```

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

Zadání: skript51.sh

Napište **skript51.sh** , který:

1. Vytvoří vždy soubor **pozdrav.txt** obsahující větu *Ahoj svete!*
2. Při spuštění s **-a** vypíše na obrazovku ahoj
3. Při spuštění s **-d** vypíše aktuální datum
4. Při spuštění s **-p** 1 2 3 vypíše počet parametrů skriptu
5. Při spuštění s jinými parametry vypíše text *Tak tedy nevim!*

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, obsahující zadaný řetězec (2.par)

./skript52.sh . truhlik

Řešení: skript52.sh

```
#!/bin/bash
if test "$#" -ne 2
then
   echo "Zadej nazev adresare a retezec, co budu hledat!"
    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 (!)

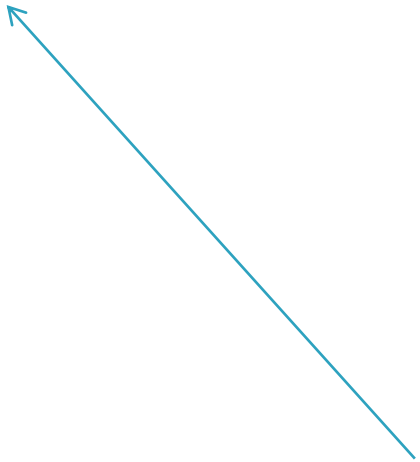
1. MOJE=“Ahoj”
   Pozor kolem znaku = nejsou mezery! Častá chyba
2. echo MOJE .. Vypíše MOJE
3. echo $MOJE .. Vypíše Ahoj
4. touch $MOJE x touch MOJE

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 zapomenu 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 bash
- man test

Funkce v bashi

```
!#/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

```
#!/bin/bash
# prevzato z http://www.abclinuxu.cz/clanky/navody/bash-iv

vrat_retezec()  {
   echo "Řetězec"
}

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

Funkce v bashi – parametry

- 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

```
#!/bin/bash

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

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

Napište jeden skript v bashi, který:

- Skript
  - vypíše Jsem spuštěný bez parametru.
- 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 (nápověda – 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

awk – manipulace s textem

- who | awk '{ print $1 }'
- who | awk '{ print $6, $1 }'

Máme text. soubor seznam1 (jmeno prijmeni)
Chceme vypsat ve tvaru (prijmeni jmeno)

- awk '{ print $2, $1}' seznam1.txt
- awk '{ print NR, $2, $1}' seznam1.txt

awk

- awk '/^Jan/ { print NR, $2,$1}' seznam1.txt
- awk '/Novak$/ { print NR, $2,$1}' seznam1.txt

- /^Jan/ .. Vezme řádky začínající na Jan
- /Novak$/ .. Řádka končí na Novák

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

Další úkoly

- Stáhněte si příkazem wget: http://home.zcu.cz/~pesicka/zos/logy

- OK
- ERROR
- CRITICAL ERROR
- FAIL

```
wget …
chmod u+x logy
./logy
./logy 5
```

1. Jedním příkazem vytvořte soubor s chybovými hláškami **chyby.txt** a soubor s ok hláškami **ok.txt**
2. Vytvořte soubor **vse.txt** s kompletním výstupem logovacího programu
3. Vypište na obrazovku jen chybové stavy, tak aby byly vypsány po stránkách
4. Vytvořte adresáře, podle druhů chyb, tzn. fail, error...

V každém adresáři vytvořte soubor chyby.txt, který bude obsahovat jen výpisy chyb daného typu

- Na konec tohoto souboru napište počet chybových hlášení
- Do souboru **proceschyby.txt** vložte čísla procesů a jejich počet chyb