Úvod do operačných systémov

Cvičenie 5

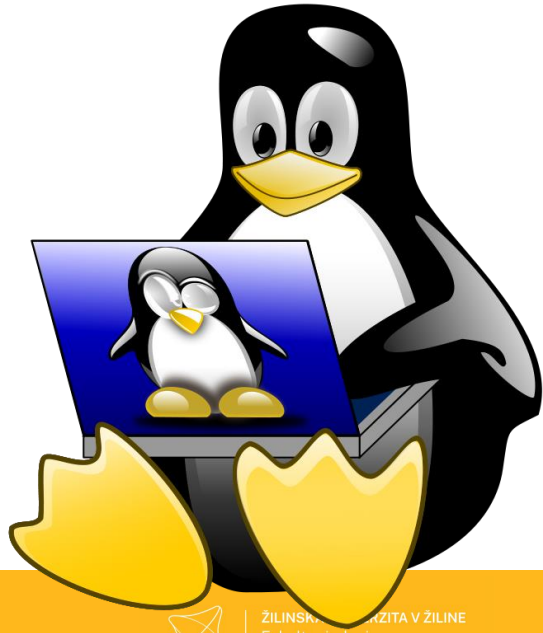
ŽILINSKÁ UNIVERZITA V ŽILINE
Fakulta riadenia
a informatiky

Náplň cvičenia

- Základy skriptovania
- Ukážka základných príkazov a ich použitie
- Polia, funkcie a návratové hodnoty

Skript a shell

- Postupnosť príkazov, ktoré sa majú vykonať a táto postupnosť je uložená v súbore
- Štandardne má tento súbor príponu .sh
- Bash - používaný shell
- Nano/Vim/Neovim - používané editory CLI
- Kate - používaný editor KDE
- Môžete použiť akýkoľvek iný softvérový nástroj



Prvý skript

- Súbor s názvom `prvy.sh` a nasledovným obsahom
  - `#! /bin/bash`
  - `echo "Ahoj $USER, dnes je $(date)"`
- Komentáre sa začínajú znakom #
  - `: ' bash umožnuje`
  - `aj viac riadkové komentáre'`
- Shebang (#!)
  - Prvý riadok, obsahuje cestu k interpreteru, ktorý daný skript interpretuje
- Spustenie pomocou nového bash-u
  - `bash prvy.sh`
- Spustenie ako spustiteľný súbor
  - `./prvy.sh`
- Nezabudnite na práva!

Načítanie hodnoty

- Načítanie prebieha príkazom `read`
  - Pokiaľ sa nešpecifikuje výstupná premenná, tak sa výsledok uloží do `REPLY`
  - Prepínačom `-p` je možné pri načítaní vypísať aj zvolenú informáciu
  - Prepínačom `-tN` je možné obmedziť zadanie hodnoty na `N` sekúnd
  - Prepínačom `-nN` je možné obmedziť počet zadávaných znakov na `N`
  - Prepínačom `-s` sa zamedzí zobrazenie napísanej hodnoty
  - `help read`
- Upravte súbor `prvy.sh` nasledovne
  - `#! /bin/bash`
  - `read -n2 -t5 -p "Zadaj svoj vek :" VEK`
  - `echo "Ahoj $USER, dnes je $(date) a mas $VEK rokov!"`

Vstupné argumenty a výstup

Označenie	Význam
`${0} - ${255}`	Prístup ku konkrétnemu argumentu
`$#`	Počet argumentov
`$@`	Zoznam argumentov okrem `$0` ako samostatné reťazce
`$*`	Zoznam argumentov okrem `$0` ako jeden reťazec
`shift N`	Posunutie spracovávaných argumentov o `N`
`set N O P`	Nastavenie argumentov `N O P`
`exit 1`	Skončenie procesu s návratovou hodnotou `1`

- Vstupné argumenty sa zadávajú pri spustení skriptu za jeho názov, pričom jednotlivé argumenty sú oddelené medzerou
  - `./prvy.sh 1 2 Ahoj svet "Ahoj svet" -x ddd ggg`
- Pri získaní hodnoty argumentu 10, 11, ... je potrebné dbať na použitie `{}`
  - `${11} != $11`

Práca s premennými

Označenie	Význam
`${#N}`	Dĺžka premennej `N`
`${N:I}`	Podreťazec hodnoty premennej `N` od pozície `I` (pri zápornej hodnote od konca, pričom je potrebné použiť `()`)
`${N:I:J}`	Podreťazec hodnoty premennej `N` od pozície `I` s dĺžkou `J`
`${N:=1}`	Hodnota premennej `N` alebo jej nastavenie na hodnotu `1` ak je nenastavená (nedefinovaná pri zadaní `:`)
`${N/A/a}`	Nahradenie prvého výskytu `A` v premennej `N` na hodnotu `a`
`${N//A/a}`	Nahradenie všetkých výskytov `A` v premennej `N` na hodnotu `a`
`${N#udos}`	Odstránenie začiatku(prefixu) `udos` z premennej `N`
`${N%.sh}`	Odstránenie konca(sufixu) `.sh` z premennej `N`
`${!P}`	Za predpokladu, že `P=N` sa vypíše hodnota premennej `N`
`${N,}` alebo `${N,,}`	Zmena prvého písmena alebo všetkých v premennej `N` na malé
`${N^}` alebo `${N^^}`	Zmena prvého písmena alebo všetkých v premennej `N` na veľké

Návratová hodnota

- Po skončení skriptu, funkcie či príkazu je možné získať návratovú hodnotu
  - `echo $?`
- Pri použití nepomenovaných dátovodov je možné získať hodnotu príkazu **n** nasledovne
  - `echo ${PIPESTATUS[n]}`
- Návratová hodnota je z rozsahu <0,255>
- Hodnota 0 predstavuje úspešné skončenie, iná hodnota ako nula znamená neštandardné skončenie

Aritmetické operácie

- +, -, *, /, %
- +=, -=, *=, /=, %=
- ++, --, **
- >, <, >=, <=, ==, !=,
- Pomocou `(( ))` alebo príkazom `let`
  - `echo $(( 2 ** 6 ))`
  - `let RESULT=2**6`

Logické operácie

- 0 - true, 1 - false
- Celé čísla: `-eq`, `-ne`, `-lt`, `-le`, `-gt`, `-ge`
- Reťazce: `-z`, `-n`, `=`, `!=`, `=~`, `<`, `>`
- Súbory: `-r`, `-w`, `-x`, `-f`, `-d`, `-e`, `-s`, `-ef`, `-nt`, `-ot`
- `&&`, `-a`, `||`, `-o`, `!`
- Pomocou `[ ]`,`[[ ]]` (nemusí byť kompatibilné s inými shell-mi) alebo príkazom `test`
  - `[ -f prvy.sh ]`
  - `test $A -lt 10 -a $A -ge 0`

Vetvenie

Príkaz `if`

- Princíp podobný ako v iných programovacích jazykoch
- Ukážka zápisu príkazu:

```
if podmienka; then
   prikazy
elif podmienka2
then
   prikazy2
   ...
else
   prikazyN
fi
```

Príkaz `case`

- Princíp podobný ako v prípade príkazu switch z iných programovacích jazykoch, `default` je nahradený znakom * a každá možnosť je ukončená znakmi `;;`
- Ukážka zápisu príkazu:

```
case hodnota in
   vzor1) prikazy ;;
   vzor2) prikazy2 ;;
   ...
   *) príkazyN ;;
esac
```

Vetvenie

```
#! /bin/bash
read -p "Zadajte cislo:"
if [ $REPLY -eq 0 ]; then
        echo "Zadali ste 0!"
        exit 0
elif [ $REPLY -lt 0 ]
then
        echo "Zadali ste hodnotu mensiu ako 0!"
        exit 1
else
        echo "Zadali ste hodnotu vacsiu ako 0!"
fi
exit 2
```

```
#! /bin/bash
vysledok=0
read -t5 -p "Zadajte heslo do 5 sek:" HESLO
case $HESLO in
kreslo|KRESLO|Kreslo)
        echo -e "Spravne heslo, vitajte $USER!"
        ;;
heslo)
        echo "Takto by to neslo!"
        vysledok=1
        ;;
*)
        echo "Neplatne heslo!"
        vysledok=2
        ;;
esac
exit $vysledok
```

Cykly

Cyklus `for`

- Zodpovedá cyklu `for` z iných programovacích jazykov
- Ako množinu je potrebné uviesť zoznam hodnôt
  - `seq 1 5`, `{1..5}`, `1 2 3 4 5`, ...
- Príklady zápisu

```
for i in množina
do
   prikazy
done

for (( i=0; i < 10; i++));do
   prikazy
done
```

Cyklus `while`

- Zodpovedá cyklu `while` z iných programovacích jazykov
- Pokiaľ podmienka platí, tak vykonávaj (opačne funguje príkaz `until`)
- Príklad zápisu

```
while podmienka
do      prikazy
done
```

Cykly

```
#! /bin/bash
for i in $@; do
        echo $i
done
for i in $(seq 1 15)
do
        echo $i
done
for i in {A..F}
do      echo $i
done
for (( i = 1; i <= 10; i++))
do
        echo $i
done
exit 0
```

```
#! /bin/bash
let A=3**2
until [ $A -le 0 ];do
        echo $A
        ((A--))
done
while test $A -ne 10
do
        echo $A
        let A++
done
exit 0
```

Cykly

Príkaz `select`

- Tento príkaz umožňuje sprostredkovať výber z daných možností
- Prompt pre výber je uložený v premennej **PS3**
- Ako množinu je potrebné uviesť zoznam hodnôt, ktoré je možné vybrať
- Príklad zápisu

```
select volba in množina
do
   prikazy
done
```

```
#!/bin/bash
let A=5
let B=6
PS3="Vyberte si operaciu: "
select op in + - "*" / %
do
        case $op in
        +|-|"*"|/|%)
                echo "$A $op $B = $(($A $op $B))" ;;
        *)
                echo "Koniec vyberu operacie"
                break ;;
        esac
done
exit 0
```

Funkcie

- Príkaz `function`
- Návratová hodnota je nepovinná a je vrátená príkazom `return N`, pričom `N` môže mať len celočíselnú hodnotu od `0` po `255`
- Parametre funkcie sú zadávané ako argumenty skriptu a pri definícii funkcie sa neuvádzajú do zátvoriek
- Vo funkcii je možné používať lokálne premenné – príkaz `local`
- Ukážky použitia:

```
function nazov_funkcie
{
   prikazy
   return n
}
```

```
nazov_funkcie()
{
   prikazy
   return n
}
```

Funkcie

```
#! /bin/bash
sucet() {
        echo $(($1 + $2))
        return 0
}
function rozdiel
{
        echo $(($1 - $2))
}
if test $# -ge 2; then
        temp=$(sucet $1 $2)
        echo "sucet je $temp"
        temp=$(rozdiel $1 $2)
        echo "rozdiel je $temp"
fi
exit 0
```

```
#! /bin/bash
mocnina() {
        local x=0
        read -p "Zadaj cele cislo: " x
        echo $((x**2))
        return 0
}
vysledok=$(mocnina)
echo "Mocnina zadaneho cisla je $ vysledok"
echo "Test lokalnej premennej x = $x"
exit 0
```

Polia

- Podobne ako v iných programovacích jazykoch aj v bash-i je možné používať polia
- Polia bývajú dvoch druhov: indexované a asociatívne (nazývané aj dictionaries)
  - Indexované sú také, pri ktorých sa k prvkom poľa pristupuje cez číselnú hodnotu indexu
  - Asociatívne sú také, pri ktorých sa k prvkom poľa pristupuje pomocou kľúča, ktorý nemusí mať celočíselnú hodnotu
- Prístup ku konkrétnemu prvku poľa `N` cez `N[index/kluc]`
- Všetky prvky je možné získať cez `${N[*]}` a cez `${N[@]}` (rozdiel ako pri argumentoch)
- Získanie indexu/kľúča pomocou `!` (bang), napr. `${!N[@]}`
- Pridanie ďalších prvkov do poľa je možné cez operátor `+=(dalsie_prvky)` alebo cez `N[5]=nova_hodnota`
- Počet prvkov v poli `N` `${#N[*]}`
- Vymazanie prvku poľa alebo celého poľa pomocou príkazu `unset N[3]` alebo `unset N`

Polia

Indexované pole

- Je možné ho vytvoriť pomocou príkazu `declare -a N` alebo priradením hodnôt `N=(jeden dva tri)`
- Získanie hodnoty `${N[3]}`
- Zmena hodnoty `N[3]=štyri`
- Pridanie viacerých hodnôt `N+=(päť šesť)`
- Odstránenie hodnoty `unset N[5]`

Asociatívne pole

- Je možné ho vytvoriť pomocou príkazu `declare -A N`
- Získanie hodnoty `${N[one]}`
- Zmena hodnoty `N[three]=tri`
- Pridanie viacerých hodnôt `N+=([four]=štyri [five]=päť)`
- Odstránenie hodnoty `unset N[five]`

Polia

```
#! /bin/bash
pole=(jeden dva tri)
pole+=(styri pat "sest sedem")
declare -a pole2
pole2[0]='one'
pole2[2]='three'
pole2[1]='two'
echo ${pole[*]}" "${pole2[@]}
unset pole[4]
for i in "${pole[@]}"
do      echo $i
done
for i in "${pole2[*]}"; do
        echo $i
done
exit 0
```

```
#! /bin/bash
declare -A pole
pole=([jeden]=one [dva]=two [3]=three)
pole[styri]=4
echo "Pocet prvkov pola = "${#pole[@]}
echo "Prvky = "${pole[*]}
echo "Prvky cez cyklus for:"
for i in "${pole[@]}"
do      echo $i
done
echo "Kluce v poli"
for i in "${!pole[@]}"; do
        echo $i
done
unset pole
echo "Prvky = "${pole[*]}
exit 0
```

Príklady na precvičenie

- Vytvorte skript, ktorý vyzve používateľa na zadanie roku a následne vypíšte počet rokov (celé číslo) medzi zadaným rokom a aktuálnym rokom. Pokiaľ používateľ nezadá rok do 5 sekúnd, program miesto toho vypíše aktuálny čas (hodiny a minúty)
- Vytvorte skript, ktorý vypíšte počet všetkých vstupných parametrov, vypíše hodnotu 1., 5. a 11. argumentu, následne vypíše veľkosť 2 argumentu (počet znakov daného argumentu). Potom nastaví argumenty 1,2, tri, four, something ako argumenty skriptu a pre piaty argument vypíšte podreťazec od pozície 4 s dĺžkou 5. Po skončení vráti skript hodnotu 1
- Vytvorte skript, ktorý získa od používateľa 2 celé čísla a urobí ich +,−,*,/ (ak je to možné vykonať), % a **
- Vytvorte skript, ktorý vygeneruje 3 náhodné čísla v rozsahu 1-9 a následne počká od používateľa na zadanie čísla z rozsahu 1-9. Pokiaľ zadané číslo bolo vygenerované, tak vypíšte gratulácia, vyhrali ste 1000€ (pokiaľ bolo vygenerovaných viac daných čísel, tak cena bude toľkokrát väčšia, t.j. pri 2 bude 2000 a pri 3 3000). Ak nie, tak vypíše smola, veľa šťastia nabudúce

Príklady na precvičenie

- Vytvorte skript, pomocou ktorého budete simulovať hru myslím si číslo, pričom používateľ bude mať za úlohu uhádnuť číslo. Vytvorte tiež skript, pomocou ktorého budete simulovať hru myslím si číslo, pričom používateľ bude mať za úlohu zvoliť náhodné číslo a skript ho bude musieť uhádnuť
- Vytvorte skript, v ktorom hodíte y-hrannou kockou x-krát (x a 3<y<31 sú hodnoty zadané ako argument skriptu, pričom ak nie sú zadané alebo nie sú platné, tak skript sa hneď skončí) a následne sa vypíše, koľko krát padli jednotlivé hodnoty na kocke
- Vytvorte skript, ktorý od používateľa bude požadovať cestu. Následne v danom adresári nájde všetky súbory používateľa skriptu, pričom pokiaľ je daný súbor adresárom, tak mu zobrazí aj obsah daného adresára a pokiaľ by bol nájdený súbor prázdny, tak mu bude ponúknuté príslušný súbor vymazať. Pokiaľ používateľ zadá miesto adresára súbor, tak sa vypíše počet riadkov, slov, znakov v súbore, ak je tento súbor textový. Ak sa jedná o binárny súbor, tak sa nevykoná nič