Operativni sistemi I

Vežbe 7

OSNOVE SHELL PROGRAMIRANJA-NASTAVAK

Shell proširenja (Shell Expansions)

Proširenje se izvršava na komandnoj liniji. Bash prepoznaje sedam proširenja i izvršava ih sledećim redom:

- proširenje preko zagrada (brace expansion) – zamena sadržaja u zagradama
- tilda proširenje (tilde expansion) – tilda zamena
- proširenje parametara i promenljivih (parameter and variable expansion) zamena parametara i promenljivih
- aritmetičko proširenje (arithmetic expansion)
- zamena komandi (command substitution), koja se obavlja sleva nadesno
- razdvajanje reči (word splitting)
- proširenje imena datoteke (filename expansion).

Samo proširenje u zagradama, razdvajanje reči i proširenje imena datoteka mogu promeniti broj reči u proširenju, ostala proširenja proširuju jednu reč u jednu reč.

Proširenje preko zagrada (Brace Expansion)

Proširenje preko zagrada je mehanizam kojim se mogu proširiti proizvoljni nizovi. Ovaj mehanizam je sličan proširenju imena datoteke, ali generisana imena datoteka ne moraju da postoje. Uzorci koji se preko zagrada proširuju uzimaju formu opcionog uvodnog dela, koju prati serija zapetom razdvojenih nizova između para zagrada, iza kojih ide opcioni dotatak. Uvodni deo je prefiks svakog niza koji se nalazi unutar zagrada, a dodatak se dodaje s desne strane na svaki rezultujući niz.

Proširenja preko zagrada mogu da se se umeću jedno u drugo. Rezultati svakog proširenog niza nisu sortirani, samo se poštuje poredak sleva nadesno, odnosno prefiks, zatim niz iz zagrade, i na kraju dodatak-sufiks.

Primer.

Proširenje komande **echo**:

```
$ echo a{d,c,b}e
ade ace abe
```

Proširenje preko zagrade se izvršava pre bilo kog drugog proširenja. Bilo koji karakter koji ima specijalno značenje za ostala proširenja čuva se u rezultatu, odnosno ne dira se. To je strogo tekstualno proširenje.

Bash ne primenjuje interpretaciju u kontekstu proširenja ili teksta između zagrada. Da bi izbegavao

konflikte sa parametarskim proširenjima niz "${" se ne smatra pogodnim za proširenje preko zagrada. Korektno formirano proširenje preko zagrada mora sadržati otvorenu i zatvorenu zagradu koje su van navodnika, i barem jednu zapetu. Svako nekorektno proširenje se ne izvršava.

Ova konstrukcija se tipično koristi kao skraćenica kada se isti zajednički prefiks generiše više puta. Tako se:

Primer.

```
$ mkdir /home/jsmith/{data,video,mp3}
```

proširuje u:

```
$ mkdir /home/jsmith/data
$ mkdir /home/jsmith/video
$ mkdir /home/jsmith/mp3
```

Komplikovaniji slučaj je korišćenje ugnježdenih proširenja.

```
$ chown root /home/{jsmith/{ss1,ss2},nmacek/{data,ss3}}
```

proširuje se u:

```
$ chown root /home/jsmith/ss1
$ chown root /home/jsmith/ss2
$ chown root /home/nmacek/data
$ chown root /home/nmacek/ss3
```

Tilda proširenje (Tilde Expansion)

Ako reč počinje tilda karakterom koji nije pod navodnicima (~), svi karakeri do prve kose crte koja je takođe van navodnika (/ slash) tretiraju se kao tilda prefiks. Ukoliko nema kose crte onda su svi karakteri tilda prefiks.

Ukoliko nema karaktera pod navodnicima unutar tilda prefiksa, tilda prefiks se tretira kao potencijalno ime korisnika za login proceduru (login-name). Tilda prefiks se zamenjuje po sledećim pravilima:

ako je login-name nulte dužine, tilda se zamenjuje vrednošću HOME promenljive, a ako je HOME promenljiva nepostavljena, tilda se zamenjuje home direktorijumom korisnika koji izvršava taj komandi interpreter.

U drugom slučaju tilda prefiks se zamenjuje home direktorijumom specificiranog korisnika (login-name). Ako je vrednost tilda prefiksa "~+", tada tilda prefiks uzima vrednost shell promenljive PWD koja predstavlja tekući radni direktorijum.

Ako je vrednost tilda prefiksa "~-", tada tilda prefiks uzima vrednost shell promenljive OLDPWD koja predstavlja prethodni tekući radni direktorijum (pod uslovom da je OLDPWD setovana).

Ako je login-name pogrešan, tilda proširenje se ne izvršava, reč s leve stane ostaje nepromenjena. Svaka dodela promenljivoj se proverava za tilda prefikse van navodnika iza kojih neposredno ide : ili =. U ovim slučajevima tilda proširenje se takođe izvršava. Prema tome, nekom mogu koristiti imena datoteka sa tildom u dodeljivanju sistemskih promenljivih kao što je PATH, MAILPATH i CDPATH, a komandni interpreter će im dodeliti proširene vrednosti.

Upotreba tilda proširenja za pozicioniranje na home direktorijum:

```
~         vrednost promenljive $HOME (/home/jsmith)
~         /data $HOME/data (/home/jsmith/data)
~jim      home directorijum korisnika jim (/home/jim).
```

```
Sledeći primer demonstrira upotrebu tilda proširenja za promenljivu $OLDPWD:
~+/misc   $PWD/misc
~-/temp   $OLDPWD/temp
```

Primer 1.

```
$ whoami
jsmith
$ pwd
/etc
$ cd ~/data     # poddirektorijum data home direktorijuma
$ pwd
/home/jsmith/data
$ cd ~jim       # home direktorijum korisnika jim
$ pwd
/home/jim
```

Primer 2.

```
$ pwd
/etc
$ cd /bin
$ cd ~-/network
$ pwd
/etc/network
```

Komandna zamena (Command Substitution)

Komandna zamena dozvoljava da se izlaz komande zameni samom komandom, odnosno da izlaz jedne komande postaje argumenat druge. Komanda zamena se izvršava kada se komanda zatvori zagradama ili navodnicima, kao u sledećim primerima:

```
$(command)
```

ili

```
`command`
```

Bash izvršava proširenje izvršavanjem komande command i zamenjuje komandnu substituciju sa standardnim izlazom komande. Ugrađene nove linije se ne brišu, ali mogu da se uklone za vreme razdvajanja reči.

Kada se koristi zamena stilom forme obrnutog navodnika, karakter obrnuta kosa crta zadržava doslovno značenje osim kada je praćen sa "$", "`", ili "\". Prvi obrnuti navodnik, koji nije praćen obrnutom kosom crtom, prekida komandnu zamenu.

Kada se koristi **$(command)** forma, svi karakteri između malih zagrada tretiraju se kao komande,

ništa se ne tretira specijalno. Ako se zamena pojavljuje sa duplim navodnicima, razdvajanje reči i proširenje imena datoteka datoteka se ne izvršava.

Primer.

Pronalaženje svih datoteka sa ekstenzijom bak.

```
$ find / -name '*.bak' –print
```

Komprimovanje istih u jednoj komandi može se izvršiti na dva načina:

```
$ gzip ` find / -name '*.bak' –print `
```

ili

```
$ gzip $( find / -name '*.bak' –print )
```

Dodatno, pomoću komandne zamene se mogu dodeliti vrednosti promenljivama:

```
$ x = `date`
$ echo $x
Thu Apr 15 09:53:44 CEST 2004
$ y = `who am i;pwd`
$ echo $y
nmacek pts/0 Apr 15 09:40 (nicotine.internal.vets.edu.yu)
/home/nmacek
```

Aritmetičko proširenje (Arithmetic Expansion)

Aritmetičko proširenje omogućava izračunavanje aritmetičkog izraza i zamenu rezultata. Format aritmetičkog izraza je:

```
$(( expression ))
```

ili

```
$[ expression ]
```

Izraz se tretira kao da je bio u duplim navodnicima, ali dupli navodnici unutar zagrada se ne tretiraju specijalno. Svi simboli u izrazu podležu parametarskom proširenju, komandnoj zameni i navodničkom uklanjanju. Aritmetičke zamene mogu da se gnezde.

Izračunavanje se izvršava prema pravilima shell aritmetike. Ako je izraz pogrešan bash prikazuje poruku koja prijavljuje otkaz i zamena se ne izvršava.

Evo nekoliko primera:

```
$ echo 1 + 1          # shell interpretira 1 + 1 kao string
1 + 1
$ echo $((1+1))       # $((1+1)) je aritmetiko proširenje
2
$ echo $((7/2))       # bash koristi celobrojnu aritmetiku
3
$ echo 3/4|bc -l            # celobrojna aritmetika
0.75
$ a=1
$ b=2
$ echo $(($a+$b))     # promenljive i aritmetiko proširenje
3
```

Bash koristi celobrojnu aritmetiku - komanda **echo $[3/4]** na ekranu prikazuje 0. Ukoliko je potrebno izvršiti neku operaciju sa realnim rezultatom ili više matematičkih operacija, može se koristiti program bc - rezultat komande **echo 3/4 | bc -l** je 0.75, što je korektno.

Aritmetičko proširenje se može iskoristiti za određivanje istinitosti izraza. U tom slučaju, proširenje vraa status 0 ili 1 zavisno od vrednosti uslovnog izraza expression. Izraz se komponuje pomoću operatora <, <=, >, >=, == i ! =. Dodatno, izrazi mogu da se kombinuju koristeći sledeće operatore:

- **(expression)** vraća vrednost izraza expression.

- **! expression** tačno ukoliko je expression netačan (negacija)
- **exp1 && exp2** tačno samo pod uslovom ako su oba izraza (exp1 i exp2) tačni
- **exp1 || exp2** tačno ako je bar jedan od izraza (exp1 ili exp2 tačan).

Operatori && i || ne izračunavaju vrednost exp2 ako je vrednost izraza exp1 dovoljna da odredi povratnu vrednost celog uslovnog izraza.

```
$ echo $((1>3||2<4))
1
$ echo $((1>3&&2==2))
0
```

Kada se koriste operatori "==" i "!=" niz desnog operatora smatra se uzorkom, a provera identičnosti odgovara pravilima za pronalaženje uzorka (Pattern Matching). Vrednost 0 se vraća ako niz odgovara uzorku, a vrednost 1 ako ne odgovara. Razdvajanje reči i proširenje imena datoteka se ne izvršavaju unutar ovog proširenja; tilda proširenje, parametarsko proširenje, komandna zamena, procesna zamena i upotreba navodnika se izvršavaju.

```
$ ime=jsmith
$ echo $(($ime==jsmith))
1
$ echo $(($ime!=jsmith))
0
```

Proširenje imena datoteka (Filename Expansion)

Nakon zadavanja komande, Bash razdvaja reči koje predstavljaju parametre i u parametrima koji predstavljaju datoteke traži karaktere "*", "?", i "[". Ako se jedan od tih karaktera pojavi tada se reč smatra uzorkom i zamenjuje se alfabetski sortiranom listom imena datoteka koja odgovara uzorku. Ukoliko je Bash pokrenut sa parametrom -f ova zamena se ne izvršava.

Pronalaženje uzorka (Pattern Matching)

Prilikom pronalaženja uzorka specijalni karakteri imaju sledeće značenje:

- ***** odgovara bilo kom nizu uključujući i niz nulte dužine. Takođe, na primer, komanda ls * prikazati sve datoteke, ls a* sve datoteke čije ime počinje sa a, a ls *.c sve datoteke koje imaju ekstenziju .c;
- **?** odgovara bilo kom karakteru. Takođe, na primer, ls ? prikazati sve datoteke čije ime ima tačno jedan karakter, a ls fo? sve datoteke čije ime ima tačno tri karaktera, od kojih su prva dva fo;
- **[...]** odgovara jednom od karaktera koji je naveden između zagrada. Ukoliko je prvi karakter iza otvorene zagrade "!" ili "^" tada odgovaraju svi karakteri koji nisu navedeni između zagrada. Na primer, ls [abc]* će prikazati sve datoteke čije ime počinje slovima a,b ili c, a ls [^abc]* sve datoteke čije ime ne počinje tim slovima;
- **[..-..]** par karaktera razdvojen znakom "-" označava zonu, odnosno opseg. Ukoliko je prvi karakter iza otvorene zagrade "!" ili "^" tada odgovaraju svi karakteri koji ne pripadaju opsegu. Na primer, ls /bin/[a-f]* će prikazati sve datoteke direktorijuma /bin čije ime počinje slovima a,b,..f, a ls /bin/[^a-e]* sve datoteke direktorijuma /bin čije ime ne počinje tim slovima;

Shell funkcije

Za deklarisanje funkcije, *bash* koristi sledeću sintaksu:

```
ime_funkcije (argumenti) { komande; }
```

Ime funkcije je praćeno zagradama. Funkcija mora biti deklarisana pre nego što nastupi bilo koji njen poziv.

Primer.

```
# func
# A simple function
repeat() {
echo -n "Ne poznajem $1 $2 "
}
repeat Pera Peric
```

Izlaz na ekran je sledeći:

```
Ne poznajem Pera Peric.
```

Kao što se vidi, argumenti se funkciji prenose isto kao što se to čini sa argumentima komandne linije, tj. unutar funkcije $1 $2... predstavljaju prvi, drugi, ... argument sa kojima je f-ja pozvana.

Povratne vrednosti

Da bi se eskplicitno postavio izlazni status funkcije, potrebno je pozvati komandu **return**:

```
return code
```

Vežba 1

Koristeći **if** izraze, napisati skript fajlskript.sh koji će:

- Uzimati jedan argument s komandne linije. Taj argument treba da bude neki direktorijum. Ako argument nije naveden, uzeće se tekući direktorijum kao podrazumevani.
- Listati sve fajlove koji imaju ekstenziju **txt**.
- Pored listanja fajlova, korisniku interaktivno treba dati da bira šta želi da se prikaže o datom fajlu, veličina, dozvole, vlasnik, grupa ili sve zajedno.

```
#!/bin/bash
################################################################################
# fajlskript.sh
# # Dec. 9, 2004
## Interaktivna skripta za informacije o tekstualnim fajlovima
## CHANGELOG:
## 12/9/04 -- This is version 1. No changes at this point
################################################################################

# Uzmi direktorijum s komandne linije i proveri da li postoji
# Ako nema argumenta ili to nije direktorijum, uzmi tekuci direktorijum
if [ -d $1 ]
then
    DIR=$1
else
    DIR=`pwd`
fi

cd $DIR

# Pita korisnika za opciju
echo "Sta zelite da znate o fajlovima: (velicina, dozvole, vlasnik, grupa,
sve)?"
read OPCIJA

if [ $OPCIJA = "velicina" ]
then
    ls -la *.txt | awk '{print $8 " : " $5}'
elif [ $OPCIJA = "dozvole" ]
then
    ls -la *.txt | awk '{print $8 " : " $1}'
elif [ $OPCIJA = "vlasnik" ]
then
    ls -la *.txt | awk '{print $8 " : " $3}'
elif [ $OPCIJA = "grupa" ]
then
    ls -la *.txt | awk '{print $8 " : " $4}'
elif [ $OPCIJA = "sve" ]
then
    ls -la *.txt
else
    echo "GRESKA: Nije izabrano nista od ponudjenih opcija!"
fi
```

Ista vežba sa case izrazom:

```
#################################################################
# case $OPTION in
# "size")
# ls -la *.txt | awk '{print $8": "$5}'
# ;;
# "permission")
# ls -la *.txt | awk '{print $8": "$1}'
# ;;
# "owner")
# ls -la *.txt | awk '{print $8": "$3}'
# ;;
# "group")
# ls -la *.txt | awk '{print $8": "$4}'
# ;;
# "all")
# ls -la *.txt | awk '{print $8": "$1", "$3", "$4", "$5}'
# ;;
# *)
# echo "Must be size, permission, owner, group, or all."
# ;;
# esac
#################################################################
```

Vežba 2

Napraviti u home direktorijumu naloga na kome ste prijavljeni poddirektorijum **vezba2**. Napisati bash shell script **kopiraj.sh** koji kopira sve fajlove iz direktorijuma /etc koji u imenu sadrže “tab” u **vezba2** ali tako da ne sadrže komentare.

```
#!/bin/bash
#
# Skripta koja kopira  fajlove  /etc/*tab* u ~/vezba2, ali bez komentara
#
cd /etc
FAJLOVI=`ls *tab*`

for FAJL in $FAJLOVI;
do
    cat $FAJL | grep -E '^[^#]' > ~/vezba2/$FAJL
done
```

Vežba 3

Interaktivno ponuditi korisniku izbor svih fajlova iz **/etc** čije se ime završava slovom “b”. Kada korisnik izabere fajl, ispisati sve informacije o tom fajlu, kao i sve linije iz njega koje sadrže cifre.

```
#!/bin/bash
#
# Skripta daje korisniku da izabere neki od fajlova /etc/*b, a zatim stampa
info o tom
# fajlu i linije koje sadrze cifre.
#

cd /etc
FAJLOVI=`ls *b`
```

```
select FAJL in $FAJLOVI;
do
    ls -l $FAJL
    echo -e "\n"
    cat $FAJL | grep -E '[0-9]'
    break;
done
```