Procesiranje komandi

- Shell ima pravila po kojima procesira zadatu komandu
- Prvo se vrši skeniranje otkucane linije (komande) i shell secka otkucano na delove (tzv. tokene) pri čemu uzima za granicu između delova karaktere navedene u $IFS (po difoltu to su space, tab i newline)
- Ako se u komandi nalazi alijas on se zamenjuje originalnom komandom
- Ispituju se delovi (na primer, očekuje se da prvi deo predstavlja naziv komande)
- Potencijalno, delovi se mogu transformisati i taj proces se naziva ***shell expansion***
- Eventualno se vrši i redirekcija o kojoj će biti reči nešto kasnije
- Tek nakon svega navedenog se komanda izvršava
- Otuda sama komanda koju shell na kraju izvršava može da se razlikuje od one otkucane

Primer

- Posmatrajmo sledeća tri jednostavna primera:
  - **echo tekst primer**
  - **echo tekst primer**
  - **echo ***
- U prvom primeru, u terminalu će biti ispisan *tekst primer*
- U drugom primeru će biti ispisano isto jer se uzastopni *space* karakteri tretiraju kao samo jedan *space* karakter - očigledno je došlo do promene onoga što smo otkucali u liniji pre samog izvršenja echo komande
- U trećem primeru će biti ispisan sadržaj direktorijuma - ovde * predstavlja džoker znak koji shell interpretira kao zamenu za sve nazive u direktorijumu pa ih zato i ispisuje sve (sličnu upotrebu smo imali ranije kod brisanja fajlova tj. **rm ***) - i ovde je došlo do transformacije otkucanoga pre samog izvršenja komande
- Napomena: u trećem primeru ako je direktorijum prazan onda će biti ispisana * jer ne postoji fajl koji bi džoker * zamenio

Primer

```
ubuntu@ubuntu-VirtualBox:~$ echo tekst primer
tekst primer
ubuntu@ubuntu-VirtualBox:~$ echo tekst            primer
tekst primer
ubuntu@ubuntu-VirtualBox:~$ echo *
Desktop Documents Downloads examples.desktop Music Pictures Public Templates Vid
eos
ubuntu@ubuntu-VirtualBox:~$ echo D*
Desktop Documents Downloads
ubuntu@ubuntu-VirtualBox:~$
```

Primetiti da kod ispisa sadržaja direktorijuma ispis nije obojen.

echo D* je ispisao sve fajlove u direktorijumu koji počinju sa D. Setite se da su nazivi case sensitive pa **echo d*** ne bi ispisao ništa.

Shell expansion - tipovi

- *Brace expansion* - koriste se {} zagrade da bi se kreirale višestruke string kombinacije sa onim što se nalazi ispred i iza zagrada, a u samim zagradama se nalaze string vrednosti po kojima se kombinacije razlikuju
- *Tilde expansion* - koristi se simbol ~ koji igra ulogu putanje home direktorijuma
- *Parameter expansion* - simbol $ se koristi na početku naziva parametra; sam navedeni parametar se menja vrednošću parametra pri tumačenju komande tj. u procesu shell expansion

Shell expansion - tipovi

- *Command substitution* - koristi $ ili backtick simbol (`); ideja je da se rezultat neke druge komande koristi kao argument u tekućoj komandi
- *Arithmetic expansion* - rezultat aritmetičke operacije se koristi kao argument u tekućoj komandi; koristi se $(()) struktura za pisanje aritmetičke operacije
- *Filename expansion* - traži karaktere *,?, i [i tretira ih kao patterne za nazive fajlova i potom te karaktere menja listom fajlova koji odgovaraju patternu (tzv. *file globbing*)

Brace expansion

- Cilj ove mogućnosti je da se na jednostavan način formiraju kombinacije stringova
- Moguće je i gneždenje
- Primer upotrebe, jednostavnije kreiranje fajlova i direktorijuma koji imaju određeni pattern
- Zarezom se odvajaju susedne vrednosti
- .. se koriste za formiranje opsega brojeva ili slova

```
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
ubuntu@ubuntu-VirtualBox:~/Documents$ mkdir {2015..2016}-{0{1..9},{10..12}}
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
2015-01  2015-04  2015-07  2015-10  2016-01  2016-04  2016-07  2016-10
2015-02  2015-05  2015-08  2015-11  2016-02  2016-05  2016-08  2016-11
2015-03  2015-06  2015-09  2015-12  2016-03  2016-06  2016-09  2016-12
ubuntu@ubuntu-VirtualBox:~/Documents$
```

Brace expansion

```
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
ubuntu@ubuntu-VirtualBox:~/Documents$ touch fajl_{a..f}.txt
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
fajl_a.txt  fajl_b.txt  fajl_c.txt  fajl_d.txt  fajl_e.txt  fajl_f.txt
ubuntu@ubuntu-VirtualBox:~/Documents$ rm *
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
ubuntu@ubuntu-VirtualBox:~/Documents$ touch fajl_{f..a}.txt
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
fajl_a.txt  fajl_b.txt  fajl_c.txt  fajl_d.txt  fajl_e.txt  fajl_f.txt
ubuntu@ubuntu-VirtualBox:~/Documents$
```

Redosled niza može biti dat i u opadajućem redosledu.

Tilde expansion

- ~ igra ulogu home direktorijuma korisnika
- Ukoliko se ~ navede sama za sebe bez teksta u nastavku, tada se ~ menja vrednošću $HOME promenljive, a ako $HOME nije setovan onda se menja vrednošću home putanje ulogovanog korisnika
- Ako se iza ~ navede tekst, onda se taj tekst tretira kao ime korisnika i menjamo sve to za home putanju tog korisnika - ukoliko taj korisnik ne postoji onda se tilde expansion ne radi i sve se tretira kao običan string

Tilde expansion

```
ubuntu@ubuntu-VirtualBox:/home$ echo ~
/home/ubuntu
ubuntu@ubuntu-VirtualBox:/home$ echo $HOME
/home/ubuntu
ubuntu@ubuntu-VirtualBox:/home$ echo $home

ubuntu@ubuntu-VirtualBox:/home$ echo ~marko
/home/marko
ubuntu@ubuntu-VirtualBox:/home$ echo ~mark
~mark
ubuntu@ubuntu-VirtualBox:/home$
```

Case sensitive su i nazivi parametara

Korisnik mark ne postoji pa se ~mark tretira kao string i ne vrši se tilde expansion na njemu

Parameter expansion

- Na prethodnom slajdu smo videli i primer ove ekspanzije
- $HOME je zamenjen u echo ispisu svojom vrednošću
- Upravo to je poenta ove ekspanzije, da se parametar zameni za svoju vrednost
- $ znak ide na početak naziva parametra
- Postoje parametri shell-a poput HOME parametra, ali mogu se definisati i svoji parametri (nešto kasnije o tome)
- Takođe, ako parametar nije definisan ili nema vrednost onda se menja sa null vrednošću (isto primer na prethodnom slajdu: $home parametar koji nije definisan)

Command substitution

- Nekad je korisno da se kao argument komande upotrebi rezultat neke druge komande
- Upravo tu uskače ekspanzija *command substitution*
- Komanda se stavlja u zagradu strukture $() ili između backtick simbola ``
- Unutar zagrade se sve tretira kao deo komande tj. ne tumače se specijalni karakteri kao takvi što je razlika u odnosu na varijantu sa `
- Komande se mogu gnezditi (u slučaju upotrebe varijante sa ` ispred unutrašnjih ` treba koristiti \ kao *escape* karakter)
- **Preporuka je koristiti varijantu sa $() iz dva razloga**:
    - Čitljivije i jednostavnije za pisanje
    - Novije distribucije forsiraju upotrebu $()

Command substitution

```
ubuntu@ubuntu-VirtualBox:~$ echo $(echo tekst)
tekst
ubuntu@ubuntu-VirtualBox:~$ echo `echo tekst`
tekst
ubuntu@ubuntu-VirtualBox:~$ echo $(echo \\)
\
ubuntu@ubuntu-VirtualBox:~$ echo `echo \\`

ubuntu@ubuntu-VirtualBox:~$
```

Ovde se \ ne tumači kao poseban karakter pa prvi echo ispisuje \\ a drugi echo onda to tretira kao escape karakter i ispisuje samo jedan \.

Ovde se \ tumači kao poseban karakter pa prvi echo ispisuje \ a drugi echo onda to tretira kao escape karakter i ne ispisuje ništa. Napomena: u novijim distribucijama i varijanta sa backtickovima radi isto kao $() varijanta

Arithmetic expansion

- Ideja je da se rezultat aritmetičke operacije koristi kao argument komande, tj. da se prilikom ove ekspanzije aritmetički izraz zameni rezultatom izraza
- Mogu se koristiti samo celi brojevi tj. integer
- Aritmetički izraz se stavlja u $(()) strukturu
- Moguće je gnezditi aritmetičke izraze
- Ova ekspanzija omogućava da se shell pretvori u jednostavan kalkulator između ostaloga
- Operatori su praktično istovetni onima iz C programskog jezika poput ++, --, +, -, *, /, **, %, &, |, &&, ||,... (spisak operatora se može naći na linku http://tldp.org/LDP/Bash-Beginners-Guide/html/sect_03_04.html)
- Pored decimalnih brojeva, brojevi se mogu pisati i u drugim formatima (npr. heksadecimalni)
- Podržane su sve osnove između 2 i 64 (uključujući i njih)
- Ako se želi specificirati osnova drugačija od 10 onda se broj piše u formatu BASE#, gde BASE predstavlja osnovu, npr. 16#
- Heksadecimalni brojevi se mogu pisati sa 0x ili 0X prefiksom
- Alternativa strukturi $(()) je $[]

Arithmetic expansion

```
ubuntu@ubuntu-VirtualBox:~$ echo $((5+5))
10
ubuntu@ubuntu-VirtualBox:~$ echo $((5*(5+5)))
50
ubuntu@ubuntu-VirtualBox:~$ echo $(($((5+5))+$((2*2))))
14
ubuntu@ubuntu-VirtualBox:~$ echo $((5 && 0))
0
ubuntu@ubuntu-VirtualBox:~$ echo $((5 && 1))
1
ubuntu@ubuntu-VirtualBox:~$ echo $((5 & 1))
1
ubuntu@ubuntu-VirtualBox:~$ echo $((4 & 1))
0
ubuntu@ubuntu-VirtualBox:~$ echo $((4 | 1))
5
ubuntu@ubuntu-VirtualBox:~$ echo $((4 ** 2))
16
ubuntu@ubuntu-VirtualBox:~$ echo $((7 ** 2))
49
ubuntu@ubuntu-VirtualBox:~$ echo $((77 % 5))
2
ubuntu@ubuntu-VirtualBox:~$ echo $((77 / 5))
15
```

Gneždenje

```
ubuntu@ubuntu-VirtualBox:~$ echo $((0x5 + 0x9))
14
ubuntu@ubuntu-VirtualBox:~$ echo $((20#5 + 20#9))
14
```

Drugačije osnove

Filename expansion

- Određeni karakteri igraju ulogu specijalnih znakova u imenima fajlova
- Ova ekstenzija menja te znakove i formira konkretne nazive fajlova s kojima potom konačan oblik komande može da radi
- * predstavlja sve moguće kombinacije
- ? predstavlja bilo koji karakter (džoker za samo jedan karakter)
- [ ] predstavlja skup karaktera navedenih u zagradi - može se navoditi niz karaktera stavljajući između prvog i poslednjeg člana -
- Može se navesti i klasa karaktera ako se između [: :] stavi naziv klase poput *upper*, *digit*, *lower*, *alnum*, *ascii*... (na primer, [[:upper:]]) - na linku https://docs.racket-lang.org/guide/regexp-chars.html se mogu videti sve klase
- Ako se iza [stavi ! ili ^ onda se za pattern uzimaju svi karakteri koji nisu navedeni u zagradi tj. navedeni simboli imaju ulogu negacije

Filename expansion - primeri

```
ubuntu@ubuntu-VirtualBox:~$ ls
Desktop    Downloads         Music     Public     Videos
Documents  examples.desktop  Pictures  Templates
ubuntu@ubuntu-VirtualBox:~$ echo *
Desktop Documents Downloads examples.desktop Music Pictures Public Templates Videos
ubuntu@ubuntu-VirtualBox:~$ echo */
Desktop/ Documents/ Downloads/ Music/ Pictures/ Public/ Templates/ Videos/
ubuntu@ubuntu-VirtualBox:~$ echo ?ideos
Videos
ubuntu@ubuntu-VirtualBox:~$ echo [DP]*
Desktop Documents Downloads Pictures Public
ubuntu@ubuntu-VirtualBox:~$ echo [!DP]*
examples.desktop Music Templates Videos
ubuntu@ubuntu-VirtualBox:~$ echo [M-P]*
Music Pictures Public
ubuntu@ubuntu-VirtualBox:~$ echo [A-M]*
Desktop Documents Downloads examples.desktop Music
ubuntu@ubuntu-VirtualBox:~$ echo [[:upper:]]*
Desktop Documents Downloads Music Pictures Public Templates Videos
ubuntu@ubuntu-VirtualBox:~$ echo [[:lower:]]*
examples.desktop
ubuntu@ubuntu-VirtualBox:~$ echo [!A-M]*
Pictures Public Templates Videos
ubuntu@ubuntu-VirtualBox:~$ echo [![:lower:]]*
Desktop Documents Downloads Music Pictures Public Templates Videos
ubuntu@ubuntu-VirtualBox:~$ echo /home/*/Documents
/home/ubuntu/Documents
```

Napomena: ako se ne nađe nijedan fajl koji zadovoljava pattern onda će echo da ispiše sam pattern tj. neće doći do ekspanzije naziva fajla

Navodnici

- Navodnici se mogu koristiti za sprečavanje izvršavanja shell expansion-a pojedinih delova (tokena)
- Jednostruki navodnici potpuno sprečavaju izvršenje shell expansion-a tj. shell sve između jednostrukih navodnika tretira kao jedan deo (token)
- Dvostruki navodnici omogućavaju da se unutar njih karakteri $, ` (backtick) i \ tretiraju kao specijalni karakteri što znači da se neki tipovi shell expansion mogu kao takvi tretirati (parameter expansion, arithmetic expansion, command substitution)

Navodnici - primeri

```
ubuntu@ubuntu-VirtualBox:~$ echo tekst     primer
tekst primer
ubuntu@ubuntu-VirtualBox:~$ echo 'tekst     primer'
tekst     primer
ubuntu@ubuntu-VirtualBox:~$ echo "tekst     primer"
tekst     primer
ubuntu@ubuntu-VirtualBox:~$ echo '$HOME'
$HOME
ubuntu@ubuntu-VirtualBox:~$ echo "$HOME"
/home/ubuntu
ubuntu@ubuntu-VirtualBox:~$ echo '$((5+5))'
$((5+5))
ubuntu@ubuntu-VirtualBox:~$ echo "$((5+5))"
10
ubuntu@ubuntu-VirtualBox:~$ echo '*'
*
ubuntu@ubuntu-VirtualBox:~$ echo "*"
*
ubuntu@ubuntu-VirtualBox:~$
```

Dodatna napomena

- Može se desiti da neki fajl u nazivu sadrži space karakter
- Tada nastaje problem u radu sa tim fajlom jer ako ga želimo navesti kao argument neke komande nećemo uspeti jer space karakter se koristi kao delimiter između argumenata pa će delovati da nije u pitanju jedan fajl već dva ili više (zavisno na koliko mesta se pojavljuje space karakter), odnosno nikako nećemo moći pristupiti takvom fajlu
- Navodnici omogućavaju da se sve pod njima tretira kao jedna celina tj. space karakter unutar njih nije više delimiter pa se takvom fajlu može pristupiti

Primer

```
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
ubuntu@ubuntu-VirtualBox:~/Documents$ touch "novi fajl"
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
novi fajl
ubuntu@ubuntu-VirtualBox:~/Documents$ rm novi fajl
rm: cannot remove ‘novi’: No such file or directory
rm: cannot remove ‘fajl’: No such file or directory
ubuntu@ubuntu-VirtualBox:~/Documents$ rm "novi fajl"
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
ubuntu@ubuntu-VirtualBox:~/Documents$ touch 'novi fajl'
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
novi fajl
ubuntu@ubuntu-VirtualBox:~/Documents$ rm 'novi fajl'
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
ubuntu@ubuntu-VirtualBox:~/Documents$
```

Primer 2

```
ubuntu@ubuntu-VirtualBox:~/Documents$ echo $(cal)
August 2016 Su Mo Tu We Th Fr Sa 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
 20 21 22 23 24 25 26 27 28 29 30 31
ubuntu@ubuntu-VirtualBox:~/Documents$ echo "$(cal)"
    August 2016
Su Mo Tu We Th Fr Sa
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31
```

U ovom primeru se koristi komanda **cal** koja vraća kalendar. Koristi se command substitution tip shell expansion-a. U varijanti bez navodnika, komanda **cal** vraća rezultat koji predstavlja argumente **echo** komande. U vraćenom rezultatu se nalazi i karakter prelaska na novu liniju ali on je sa stanovišta bash-a tumačen kao delimiter tokena. U drugom slučaju, navodnici onemogućavaju da bash izdeli rezultat **cal** komande na delove (samim tim bash neće deliti rezultat na tokene, već je ceo rezultat jedan token) pa je samim tim i ispis kvalitetniji jer se sad ispisuje i prelazak u novi red – jer nisu odstranjeni newline karakteri. Naravno, ovde se moraju koristiti dvostruki navodnici jer ne želimo da potisnemo command substitution.

Prikaz shell expansion

- Komanda **set -x** aktivira prikaz izgleda komande koja će zaista biti izvršena (nakon eventualnog procesa shell expansion)
- U prvoj liniji iza komande će biti prikazan izgled komande koji će zaista biti izvršen
- Ovo je zgodno kad se želi videti da li se komanda zaista izvršava u obliku u kojem želimo i da li je shell expansion uradio ono što smo i hteli
- Komanda **set +x** deaktivira navedeni prikaz

Prikaz shell expansion - primeri

```
ubuntu@ubuntu-VirtualBox:~$ set -x
ubuntu@ubuntu-VirtualBox:~$ echo $HOME
+ echo /home/ubuntu
/home/ubuntu
ubuntu@ubuntu-VirtualBox:~$ echo "$HOME"
+ echo /home/ubuntu
/home/ubuntu
ubuntu@ubuntu-VirtualBox:~$ echo '$HOME'
+ echo '$HOME'
$HOME
ubuntu@ubuntu-VirtualBox:~$ echo D*
+ echo Desktop Documents Downloads
Desktop Documents Downloads
ubuntu@ubuntu-VirtualBox:~$ mkdir dir_{a..c}
+ mkdir dir_a dir_b dir_c
ubuntu@ubuntu-VirtualBox:~$ echo ~
+ echo /home/ubuntu
/home/ubuntu
ubuntu@ubuntu-VirtualBox:~$ echo $((5*(5+5)))
+ echo 50
50
ubuntu@ubuntu-VirtualBox:~$ echo [DP]*
+ echo Desktop Documents Downloads Pictures Public
Desktop Documents Downloads Pictures Public
ubuntu@ubuntu-VirtualBox:~$ echo [!DP]*
+ echo dir_a dir_b dir_c examples.desktop Music Templates Videos
dir_a dir_b dir_c examples.desktop Music Templates Videos
ubuntu@ubuntu-VirtualBox:~$ set +x
+ set +x
ubuntu@ubuntu-VirtualBox:~$ echo $HOME
/home/ubuntu
```

Kontrolni karakteri

- Komande se mogu dopuniti kontrolnim karakterima što omogućava dodatne mogućnosti u radu sa shell-om
- Kontrolni karakteri su:
  - ; - razdvajanje komandi
  - & - aktuelna komanda se stavlja u pozadinu
  - $? - izlazni kod prethodne komande
  - && - logičko I
  - || - logičko ILI
  - # - komentar
  - \ - omogućava da se specijalni karakter tumači kao običan ili ako se nalazi na kraju linije omogućava nastavak pisanja komande u sledećoj liniji

Razdvajanje komandi

- U jednoj liniji može biti napisano više komandi
- Između komandi se stavlja ;
- Sve do ; (ili između dva susedna ;) se tumači kao deo jedne komande
- Kada se okonča izvršenje jedne komande, onda se prelazi na izvršenje sledeće u nizu

```
ubuntu@ubuntu-VirtualBox:~/Documents$ echo prvi tekst; echo drugi tekst; echo treci tekst
prvi tekst
drugi tekst
treci tekst
ubuntu@ubuntu-VirtualBox:~/Documents$ touch fajl1
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
fajl1
ubuntu@ubuntu-VirtualBox:~/Documents$ rm fajl1; ls
ubuntu@ubuntu-VirtualBox:~/Documents$
```

U prvom primeru su izvršene tri **echo** komande. U drugom primeru je obrisan prethodno kreiran fajl i izvršeno listanje sadržaja direktorijuma nakon brisanja.

Stavljanje komande u pozadinu

- Ako se na kraju komande stavi & izvršenje komande će biti urađeno u pozadini, a terminal će biti vraćen korisniku na raspolaganje
- Većina komandi se izvrši veoma brzo, ali neke komande mogu da zahtevaju određeno vreme za svoje izvršenje pa ako se njihovo izvršenje ne stavi u pozadinu, korisniku nije omogućen pristup terminalu (u smislu zadavanja nove komande)

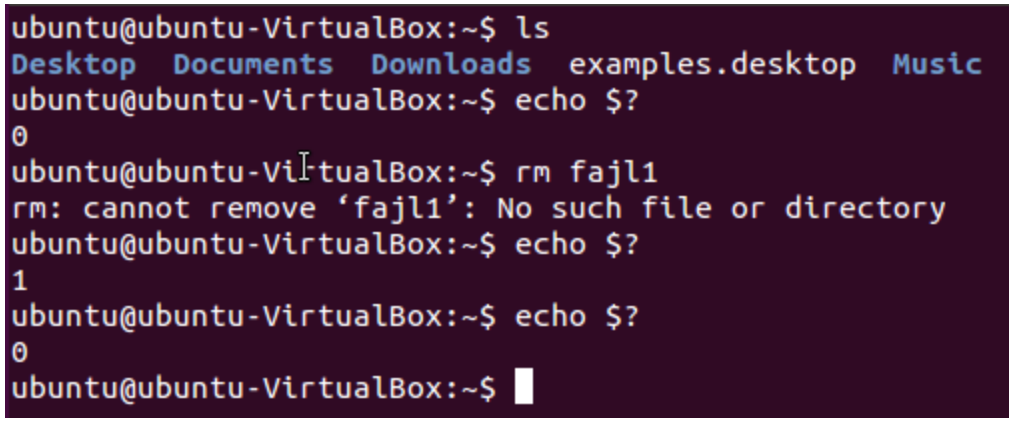
```
ubuntu@ubuntu-VirtualBox:~/Documents$ sleep 5
ubuntu@ubuntu-VirtualBox:~/Documents$ sleep 5 &
[1] 2076
ubuntu@ubuntu-VirtualBox:~/Documents$
[1]+  Done                    sleep 5
ubuntu@ubuntu-VirtualBox:~/Documents$ sleep 5 &
[1] 2077
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
ubuntu@ubuntu-VirtualBox:~/Documents$ cd ..
ubuntu@ubuntu-VirtualBox:~$ ls
Desktop  Documents  Downloads  examples.desktop  Music  Pictures
[1]+  Done                    sleep 5  (wd: ~/Documents)
(wd now: ~)
ubuntu@ubuntu-VirtualBox:~$
```

Komanda **sleep** vrši pauzu za zadato vreme. Ako se ne navedu jedinice, onda su u pitanju sekunde. Prva linija ne stavlja **sleep** u pozadinu pa je bash nedostupan narednih 5s. U drugoj liniji bash je dat na raspolaganju korisniku nakon zadavanja komande **sleep**.

U trećem primeru se vidi da se mogu zadavati druge komande dok se još uvek izvršava **sleep** u pozadini

Rezultat prethodne komande

- U shell parametar $? se stavlja izlazni kod poslednje izvršene komande



Izlazni kod 0 označava da je komanda uspešno izvršena, a 1 da nije.

Logičko I

- Dve komande mogu da se povežu sa && operatorom
- Druga komanda u nizu se izvršava samo ako se prva komanda u nizu uspešno izvršila

```
ubuntu@ubuntu-VirtualBox:~$ cd Documents && ls
fajl1  fajl2
ubuntu@ubuntu-VirtualBox:~/Documents$ cd Documents && echo Postoji ovaj dir
bash: cd: Documents: No such file or directory
ubuntu@ubuntu-VirtualBox:~/Documents$
```

U prvom primeru je izvršen prelaz u direktorijum Documents i izvršena je potom **ls** komanda na taj direktorijum. U drugom primeru je zadat ponovo prelaz u direktorijum Document koji ne postoji (neuspeh komande) i samim tim druga komanda (**echo**) nije izvršena.

Logičko I

- Može se spojiti i više komandi u niz i izvršenje svake komande zavisi od uspeha prethodnika
- Prvi neuspeh prekida lanac komandi i nakon te tačke neuspeha preostale komande se ne izvršavaju

```
ubuntu@ubuntu-VirtualBox:~$ cd Documents && ls && rm fajl1
fajl1  fajl2
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
fajl2
ubuntu@ubuntu-VirtualBox:~/Documents$
```

Dati primer je sličan primeru sa prethodnog slajda. Dodata je komanda koja briše fajl1.

Logičko ILI

- Dve komande mogu da se povežu sa || operatorom
- Druga komanda u nizu se izvršava samo ako se prva komanda u nizu nije uspešno izvršila
- I ovde može više komandi da se poveže u niz, pri čemu prvo uspešno izvršenje prekida lanac

```
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
fajl2
ubuntu@ubuntu-VirtualBox:~/Documents$ rm fajl1 || rm fajl2
rm: cannot remove ‘fajl1’: No such file or directory
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
ubuntu@ubuntu-VirtualBox:~/Documents$ cd ..
ubuntu@ubuntu-VirtualBox:~$ cd Docs || cd docs || cd Documents || cd documents
bash: cd: Docs: No such file or directory
bash: cd: docs: No such file or directory
ubuntu@ubuntu-VirtualBox:~/Documents$
```

U prvom primeru je zadato brisanje fajl koji ne postoji pa onoga koji postoji. U drugom primeru je dato izvršavanje prelaza u direktorijume, pri čemu prva dva ne postoje, a treći postoji.

Kombinovanje I i ILI

- I i ILI se mogu kombinovati u svojevrsnu IF ELSE konstrukciju
- Komanda1 && Komanda2 || Komanda3 - ako se Komanda1 uspešno izvrši onda će se još izvršiti i Komanda2, a ako se neuspešno izvrši onda će se izvršiti još komanda3

```
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
fajl1
ubuntu@ubuntu-VirtualBox:~/Documents$ rm fajl1 && echo Uspeh || echo Neuspeh
Uspeh
ubuntu@ubuntu-VirtualBox:~/Documents$ rm fajl1 && echo Uspeh || echo Neuspeh
rm: cannot remove ‘fajl1’: No such file or directory
Neuspeh
ubuntu@ubuntu-VirtualBox:~/Documents$
```

U prvom primeru brišemo fajl koji postoji pa se ispisuje **echo** koji predstavlja Komandu2 (**echo Uspeh**). U drugom primeru pokušavamo da obrišemo isti fajl (koji sada ne postoji) pa pošto dolazi do neuspeha brisanja (Komanda1) sada se aktivira Komanda3 (**echo Neuspeh**).

Komentari

- Sve iza # se tumači kao komentar tj. bash ga ne tretira kao komandu niti ga procesira
- Na primer:
  - **rm fajl1 #Brisanje fajla**
  - **cd Documents #prelaz u direktorijum Documents**
- Komentari mogu biti zgodni u edukaciji
- Takođe, pošto se komande čuvaju u bash istoriji, komentari mogu biti korisni kao podsetnik šta se radilo u terminalu i zašto su neke komande izvršavane

Izbegavanje specijalnih karaktera

- Karakter \ se može iskoristiti za izbegavanje specijalnih karaktera
- Stavljanjem \ ispred specijalnog karaktera omogućava se da se ti specijalni karakteri tumače kao obični karakteri

```
ubuntu@ubuntu-VirtualBox:~/Documents$ echo aaa # aaa
aaa
ubuntu@ubuntu-VirtualBox:~/Documents$ echo aaa \# aaa
aaa # aaa
ubuntu@ubuntu-VirtualBox:~/Documents$ echo 111; aaa
111
No command 'aaa' found, did you mean:
 Command 'aa' from package 'astronomical-almanac' (universe)
 Command 'jaaa' from package 'jaaa' (universe)
 Command 'aha' from package 'aha' (universe)
 Command 'ara' from package 'ara' (universe)
aaa: command not found
ubuntu@ubuntu-VirtualBox:~/Documents$ echo 111\; aaa
111; aaa
ubuntu@ubuntu-VirtualBox:~/Documents$
```

Komanda u više linija

- Stavljanem karaktera \ na kraj linije signalizira se da se komanda nastavlja u sledećoj liniji
- Može biti više linija (sve linije moraju da završe sa \) i prva linija koja ne završava karakterom \ predstavlja kraj komande i tada shell interpretira napisane linije kao jednu celinu

Tek kada se otkuca linija bez \ na kraju, shell interpretira komandu.

```
ubuntu@ubuntu-VirtualBox:~/Documents$ echo jedan \
> dva \
> tri
jedan dva tri
ubuntu@ubuntu-VirtualBox:~/Documents$ touch fajl1 \
> fajl2 \
> fajl3
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
fajl1  fajl2  fajl3
ubuntu@ubuntu-VirtualBox:~/Documents$
```

Shell promenljive (parametri)

- Već smo videli neke shell promenljive poput HOME
- Vrednostima promenljivih se pristupa sa $ImePromenljive - nazivi su case sensitive
- Moguće je kreirati i sopstvene promenljive
- Postoje rezervisane reči za promenljive i one ne mogu da se koriste pri kreiranju sopstvenih promenljivih (to su u stvari parametri samog shell-a poput već pomenutog HOME)
- Lokalne promenljive su one koje su vidljive samo u aktuelnoj instanci shell-a
- Globalne promenljive su one koje su vidljive u svim shell-ovima (ove promenljive se još nazivaju i promenljivima okruženja) - pod svim shell-ovima podrazumevamo originalni shell i njegove shell potomke

Shell promenljive

- Lista promenljivih sa objašnjenjima se može naći na linku http://tldp.org/LDP/Bash-Beginners-Guide/html/sect_03_02.html
- Lista globalnih promenljivih se može videti sa komandama **env** ili **printenv**
- Lista svih promenljivih se može videti sa komandom **set** (u nekim okruženjima poput Ubuntu biće ispisane i shell funkcije pa je tada zgodnije koristiti ispis u vidu **set | less**)
- Primeri promenljivih samog shell-a: HOME, USER, PATH, LANG, SHELL, TERM,....
- Podela promenljivih po tipu podataka tj. vrednosti je na string, integer, niz, konstanta.
- Napomena: sve promene setovanja koja se urade u terminalu su aktuelne za vreme sesije, ali se resetuju na staro pri ponovnom pokretanju terminala - za trajne promene potrebno je editovati konfiguracione fajlove

Primer liste

```
ubuntu@ubuntu-VirtualBox:~$ printenv
XDG_VTNR=7
XDG_SESSION_ID=c2
CLUTTER_IM_MODULE=xim
XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/ubuntu
GPG_AGENT_INFO=/run/user/1000/keyring/gpg:0:1
TERM=xterm
SHELL=/bin/bash
VTE_VERSION=3803
WINDOWID=33554443
UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-session/1000/1281
GNOME_KEYRING_CONTROL=
GTK_MODULES=overlay-scrollbar:unity-gtk-module
USER=ubuntu
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd
=40;33;01:or=40;31;01:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st
1;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lz
z4=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31
1:*.zip=01;31:*.z=01;31:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:
.lzo=01;31:*.xz=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31
:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01
1;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz
b=01;31:*.jpg=01;35:*.jpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35
:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=
01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*
.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01
1;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.as
m=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35
*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;
35:*.anx=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac
=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:
*.ra=00;36:*.wav=00;36:*.axa=00;36:*.oga=00;36:*.spx=00;36:*.xspf=00
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Session0
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
DEFAULTS_PATH=/usr/share/gconf/ubuntu.default.path
```

Prikazan je samo deo ispisa. Komanda **env** navedena bez opcija takođe daje isti prikaz tj. radi isto što i **printenv**. Ali, **env** sa opcijama omogućava i druge funkcionalnosti što ćemo videti na narednim slajdovima.

Primer liste

```
BASH=/bin/bash
BASHOPTS=checkwinsize:cmdhist:complete_fullquote:expand_aliase
:force_fignore:histappend:interactive_comments:progcomp:prompt
BASH_ALIASES=()
BASH_ARGC=()
BASH_ARGV=()
BASH_CMDS=()
BASH_COMPLETION_COMPAT_DIR=/etc/bash_completion.d
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="4" [1]="3" [2]="30" [3]="1" [4]="release"
ux-gnu")
BASH_VERSION='4.3.30(1)-release'
CLUTTER_IM_MODULE=xim
COLUMNS=80
COMPIZ_CONFIG_PROFILE=ubuntu
DBUS_SESSION_BUS_ADDRESS=unix:abstract=/tmp/dbus-Ab5mGCcLHP
DEFAULTS_PATH=/usr/share/gconf/ubuntu.default.path
DESKTOP_SESSION=ubuntu
DIRSTACK=()
DISPLAY=:0
EUID=1000
GDMSESSION=ubuntu
GDM_LANG=en_US
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
GNOME_KEYRING_CONTROL=
GNOME_KEYRING_PID=
GPG_AGENT_INFO=/run/user/1000/keyring/gpg:0:1
GROUPS=()
GTK_IM_MODULE=ibus
GTK_MODULES=overlay-scrollbar:unity-gtk-module
HISTCONTROL=ignoreboth
HISTFILE=/home/ubuntu/.bash_history
HISTFILESIZE=2000
HISTSIZE=1000
HOME=/home/ubuntu
```

Prikazan je samo deo ispisa za komandu **set | less**. Ova lista sadrži sve promenljive i lokalne i globalne. U zavisnosti od distribucije, nakon liste promenljivih može doći i do ispisa shell funkcija. U suštini komanda **set** izvršava ovaj ispis, a **less** deo omogućava samo da ispis ide stranu po stranu, umesto da sav sadržaj bude ispisan odjednom.

Pristup promenljivima

- Kao što je ranije rečeno, vrednostima promenljivih se pristupa preko $ karaktera
- Sam shell radi ekspanziju parametra (*parameter expansion*) kad naiđe na $ znak u komandi
- Shell proverava listu promenljivih i traži da li postoji promenljiva datog naziva i uzima se njena vrednost, a ako ona nema vrednost ili sama promenljiva ne postoji, uzima se null vrednost

```
ubuntu@ubuntu-VirtualBox:~$ echo $USER
ubuntu
ubuntu@ubuntu-VirtualBox:~$ echo $HOME
/home/ubuntu
ubuntu@ubuntu-VirtualBox:~$ pwd
/home/ubuntu
ubuntu@ubuntu-VirtualBox:~$ cd ..
ubuntu@ubuntu-VirtualBox:/home$ cd ..
ubuntu@ubuntu-VirtualBox:/$ pwd
/
ubuntu@ubuntu-VirtualBox:/$ cd $HOME
ubuntu@ubuntu-VirtualBox:~$ pwd
/home/ubuntu
ubuntu@ubuntu-VirtualBox:~$
```

Promenljiva HOME se može iskoristiti za navigaciju do home direktorijuma.

Kreiranje promenljive

- Kreiranje promenljive je krajnje jednostavno
- Princip je naziv promenljive = vrednost promenljive, pri čemu ne sme biti razmaka ispred i iza =
- U nazivu promenljive su dozvoljeni alfanumerički karakteri i _, ali naziv mora početi slovom

```
ubuntu@ubuntu-VirtualBox:~$ Var_Primer1=77
ubuntu@ubuntu-VirtualBox:~$ Var_Primer2="Tekst primer"
ubuntu@ubuntu-VirtualBox:~$ echo $Var_Primer1
77
ubuntu@ubuntu-VirtualBox:~$ echo $Var_Primer2
Tekst primer
ubuntu@ubuntu-VirtualBox:~$
```

Promenljivima se pristupa kao i ostalim promenljivima ($ karakter).

```
USER=ubuntu
VTE_VERSION=3803
Var_Primer1=77
Var_Primer2='Tekst primer'
```

Deo ispisa **set** komande. Kao što se vidi sada su i ove dve novokreirane promenljive prisutne.

Kreiranje promenljive

```
ubuntu@ubuntu-VirtualBox:~$ var1="$HOME"
ubuntu@ubuntu-VirtualBox:~$ var2='$HOME'
ubuntu@ubuntu-VirtualBox:~$ echo $var1
/home/ubuntu
ubuntu@ubuntu-VirtualBox:~$ echo $var2
$HOME
ubuntu@ubuntu-VirtualBox:~$
```

Ovde je prikazan uticaj dvostrukih i jednostrukih navodnika. Promenljiva var1 dobija vrednost promenljive HOME jer se koriste dvostruki navodnici. Promenljiva var2 dobija vrednost stringa $HOME jer se koriste jednostruki navodnici. Upotreba **echo** komande za ispis obe promenljive ilustruje navedeno.

Uništavanje promenljive

- Kada želimo da uništimo promenljivu koju smo ranije kreirali, koristimo komandu **unset**
- Argument ove komande je naziv promenljive koju uništavamo
- Može se navesti više promenljivih jedna za drugom

```
ubuntu@ubuntu-VirtualBox:~$ unset Var_Primer1 Var_Primer2
ubuntu@ubuntu-VirtualBox:~$ echo $Var_Primer1

ubuntu@ubuntu-VirtualBox:~$ echo $Var_Primer2

ubuntu@ubuntu-VirtualBox:~$
```

U ovom primeru su uništene promenljive kreirane u okviru primera sa prethodnog slajda. Ispis **echo** komandi je prazan jer ovih promenljivih više nema pa se one menjaju sa null u okviru parameter expansion postupka.

Ako bi se uradio ispis **set** komandom, videlo bi se da ove dve promenljive više nisu u listi.

Izvoz promenljive

- Promenljiva koju kreiramo je po difoltu lokalna promenljiva
- Ako želimo da promenljiva bude vidljiva i u drugom shell-u, neophodno je da je izvezemo (uradimo eksport promenljive)
- Komanda **export** vrši izvoz promenljive tako da bude vidljiva i u shell potomcima
- Argument **export** komande je promenljiva koju izvozimo
- Može da se navede više promenljivih odjednom

```
ubuntu@ubuntu-VirtualBox:~$ echo $SHLVL
1
ubuntu@ubuntu-VirtualBox:~$ var1="Tekst_1"
ubuntu@ubuntu-VirtualBox:~$ var2="Tekst_2"
ubuntu@ubuntu-VirtualBox:~$ var3="Tekst_3"
ubuntu@ubuntu-VirtualBox:~$ bash
ubuntu@ubuntu-VirtualBox:~$ echo $SHLVL
2
ubuntu@ubuntu-VirtualBox:~$ echo $var1 $var2 $var3

ubuntu@ubuntu-VirtualBox:~$ exit
exit
ubuntu@ubuntu-VirtualBox:~$ echo $SHLVL
1
ubuntu@ubuntu-VirtualBox:~$ export var1 var3
ubuntu@ubuntu-VirtualBox:~$ bash
ubuntu@ubuntu-VirtualBox:~$ echo $SHLVL
2
ubuntu@ubuntu-VirtualBox:~$ echo $var1 $var2 $var3
Tekst_1 Tekst_3
ubuntu@ubuntu-VirtualBox:~$
```

Tek kad su promenljive var1 i var3 izvezene, mogu se videti u shell potomku.

$SHLVL sadrži vrednost nivoa shell-a. Komanda **bash** pokreće novi shell. U primeru se može videti da je nivo novog shell-a za 1 veći (tj. to je potomak originalnog shell-a).

Izvoz promenljive

- Iz shell-a nivoa većeg broja (shell potomak) ne možemo da izvezemo promenljivu u shell nivoa nižeg broja (tj. roditeljski shell)

```
ubuntu@ubuntu-VirtualBox:~$ echo $SHLVL
2
ubuntu@ubuntu-VirtualBox:~$ var4="tekst_4"
ubuntu@ubuntu-VirtualBox:~$ echo $var4
tekst_4
ubuntu@ubuntu-VirtualBox:~$ export var4
ubuntu@ubuntu-VirtualBox:~$ exit
exit
ubuntu@ubuntu-VirtualBox:~$ echo $SHLVL
1
ubuntu@ubuntu-VirtualBox:~$ echo $var4

ubuntu@ubuntu-VirtualBox:~$
```

Komanda **exit** zatvara tekući shell i vraćamo se u roditeljski shell. Promenljiva var4 kreirana u shell detetu nije vidljiva u shell roditelju bez obzira na urađeni **export var4**.

```
ubuntu@ubuntu-VirtualBox:~$ printenv
var1=Tekst_1
XDG_VTNR=7
var3=Tekst_3
XDG_SESSION_ID=c2
```

Primetiti da su izvezene promenljive sada u listi globalnih promenljivih.

Izvoz promenljive

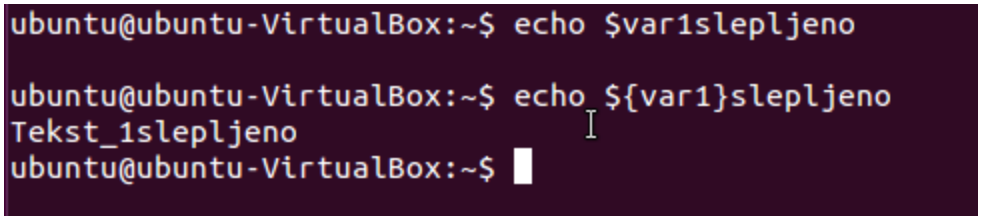
- Posledica nemogućnosti izvoženja promenljive u roditeljski shell je da sve promene urađene na promenljivoj u shell potomku neće biti prenete u roditeljski shell

```
ubuntu@ubuntu-VirtualBox:~$ echo $SHLVL
2
ubuntu@ubuntu-VirtualBox:~$ echo $var1
Tekst_1
ubuntu@ubuntu-VirtualBox:~$ var1="novi_tekst"
ubuntu@ubuntu-VirtualBox:~$ echo $var1
novi_tekst
ubuntu@ubuntu-VirtualBox:~$ exit
exit
ubuntu@ubuntu-VirtualBox:~$ echo $SHLVL
1
ubuntu@ubuntu-VirtualBox:~$ echo $var1
Tekst_1
ubuntu@ubuntu-VirtualBox:~$
```

Promenljiva var1 je promenjena u shell potomku. Ali, ta promena nije preneta u shell roditelja.

Delineacija promenljive

- Naziv promenljive shell uzima do prvog karaktera koji nije alfanumerički (ili _)
- Nekad je zgodno da iza kraja naziva promenljive ne bude karakter koji nije alfanumerički (ili _), ali tada je problem što će shell tumačiti pogrešan naziv promenljive
- Rešenje je da se naziv promenljive stavi u vitičaste zagrade {}



U prvom primeru shell traži vrednost promenljive naziva var1slepljeno, a pošto ta promenljiva ne postoji, rezultat promenljive je null pa **echo** ne ispisuje ništa. U drugom slučaju shell traži vrednost promenljive var1 koja postoji i zato je ispis **echo** komande korektan.

Nedefenisane promenljive

- Po difoltu, shell promenljive koje ne postoje zamenjuje sa null
- Ako se uradi **set -u** komanda onda će se ispisati za te promenljive da su nedefinisane (*unbounded*)
- Komanda **set +u** vraća difolt ponašanje shell-a po pitanju nedefinisanih promenljivih

```
ubuntu@ubuntu-VirtualBox:~$ echo $var5

ubuntu@ubuntu-VirtualBox:~$ set -u
ubuntu@ubuntu-VirtualBox:~$ echo $var5
bash: var5: unbound variable
ubuntu@ubuntu-VirtualBox:~$ set +u
ubuntu@ubuntu-VirtualBox:~$ echo $var5

ubuntu@ubuntu-VirtualBox:~$
```

```
ubuntu@ubuntu-VirtualBox:~$ cd $var5
ubuntu@ubuntu-VirtualBox:~$ set -u
ubuntu@ubuntu-VirtualBox:~$ cd $var5
bash: var5: unbound variable
ubuntu@ubuntu-VirtualBox:~$ set +u
ubuntu@ubuntu-VirtualBox:~$
```

Niz

```
ubuntu@ubuntu2004:~$ niz=(aa bb cc)
ubuntu@ubuntu2004:~$ echo $niz
aa
ubuntu@ubuntu2004:~$ echo ${niz[*]}
aa bb cc
ubuntu@ubuntu2004:~$ echo $niz[*]
aa[*]
ubuntu@ubuntu2004:~$ echo ${niz[2]}
cc
ubuntu@ubuntu2004:~$ echo ${niz[0]}
aa
ubuntu@ubuntu2004:~$ echo ${#niz}
2
ubuntu@ubuntu2004:~$ unset niz[1]
ubuntu@ubuntu2004:~$ echo ${niz[*]}
aa cc
ubuntu@ubuntu2004:~$ echo ${niz[1]}

ubuntu@ubuntu2004:~$ echo ${niz[2]}
cc
ubuntu@ubuntu2004:~$ niz[1]=bb
ubuntu@ubuntu2004:~$ echo ${niz[1]}
bb
ubuntu@ubuntu2004:~$
```

Upotrebom () se može kreirati niz

Pristup određenom članu ide preko []

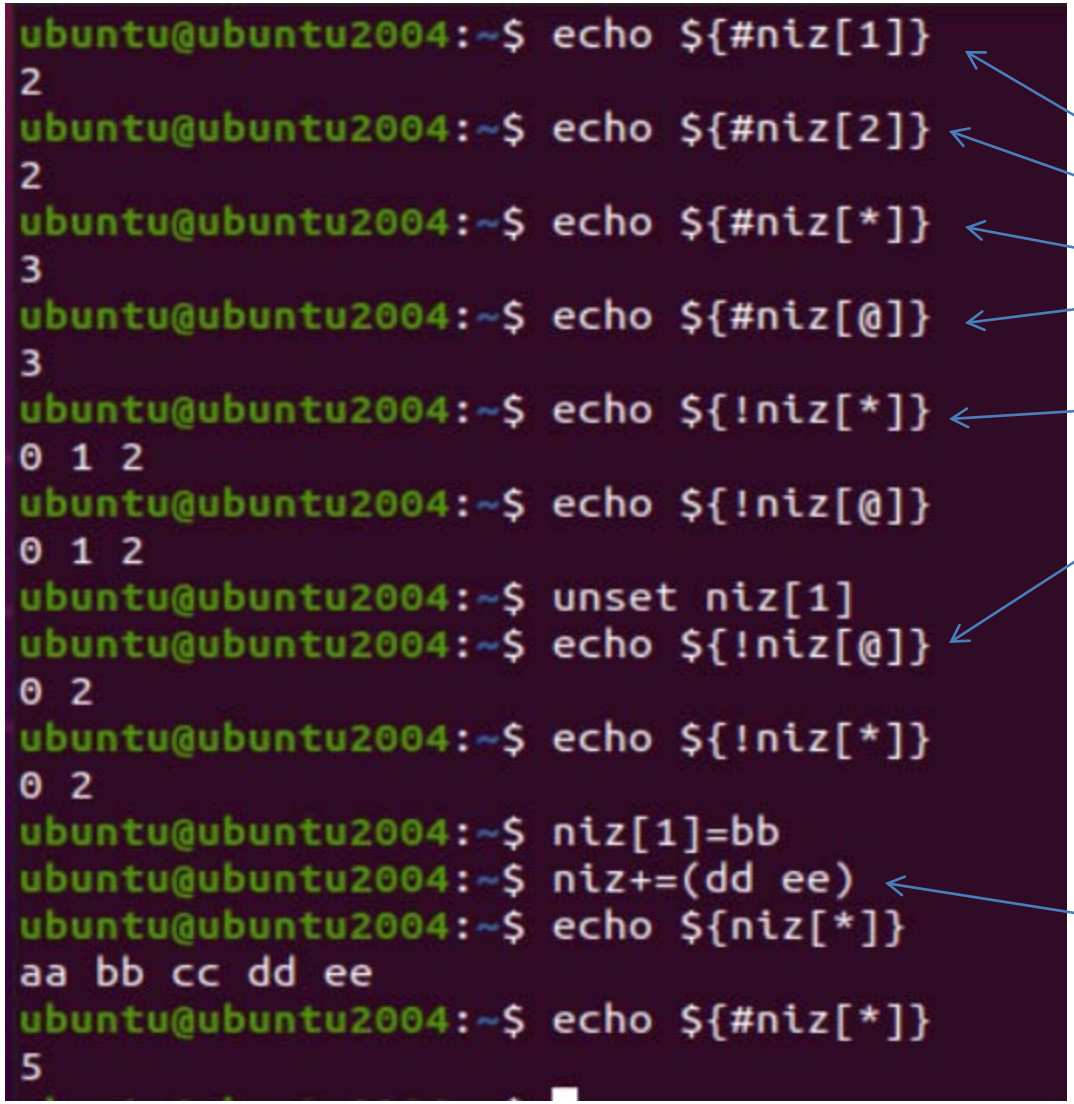
Svim članovima sa [*] ili sa [@]

```
ubuntu@ubuntu2004:~$ echo ${niz[*]}
aa bb cc
ubuntu@ubuntu2004:~$ unset niz
ubuntu@ubuntu2004:~$ echo ${niz[*]}

ubuntu@ubuntu2004:~$
```

unset uništava ceo niz tj. varijablu

Niz



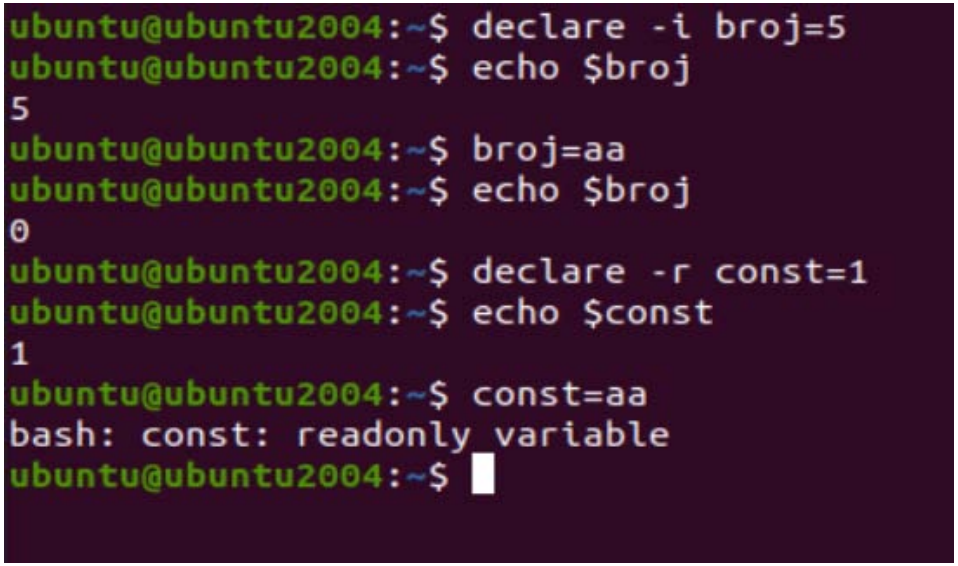
se može iskoristiti za određivanje dužine člana niza ili čitavog niza

! se može iskoristiti za ispitivanje koji indeksi niza imaju setovanu vrednost

+= se može iskoristiti za proširenje niza tj. dodavanje novih članova

Konstanta

- **declare** se može koristiti za specificiranje varijable
- Opcija -i definiše da je u pitanju integer
- Opcija -r definiše da je u pitanju read-only varijabla (konstanta)
- Spisak na https://linux.die.net/Bash-Beginners-Guide/sect_10_01.html



Shell gneždenje (embedding)

- Command substitution u stvari otvara shell potomak
- Moguće je i gneždenje command substitution struktura

```
ubuntu@ubuntu-VirtualBox:~$ $(var1="tekst";echo $var1)
No command 'tekst' found, did you mean:
 Command 'test' from package 'coreutils' (main)
tekst: command not found
ubuntu@ubuntu-VirtualBox:~$ echo $(var1="tekst";echo $var1)
tekst
ubuntu@ubuntu-VirtualBox:~$ echo $var1

ubuntu@ubuntu-VirtualBox:~$ echo `var1="tekst";echo $var1`
tekst
ubuntu@ubuntu-VirtualBox:~$ echo $var1

ubuntu@ubuntu-VirtualBox:~$ echo $(var1="gnezdo1";echo $var1 $(var2='gnezdo2';echo $var2))
gnezdo1 gnezdo2
ubuntu@ubuntu-VirtualBox:~$
```

Shell gneždenje (embedding)

```
ubuntu@ubuntu-VirtualBox:~$ $(var1="tekst";echo $var1)
No command 'tekst' found, did you mean:
 Command 'test' from package 'coreutils' (main)
tekst: command not found
ubuntu@ubuntu-VirtualBox:~$ echo $(var1="tekst";echo $var1)
tekst
ubuntu@ubuntu-VirtualBox:~$ echo $var1

ubuntu@ubuntu-VirtualBox:~$ echo `var1="tekst";echo $var1`
tekst
ubuntu@ubuntu-VirtualBox:~$ echo $var1

ubuntu@ubuntu-VirtualBox:~$ echo $(var1="gnezdo1";echo $var1 $(var2='gnezdo2';echo $var2))
gnezdo1 gnezdo2
ubuntu@ubuntu-VirtualBox:~$
```

Primetiti da je u originalnom shell-u uvek korišćen **echo**, sem u prvoj liniji koja je i proizvela grešku. Rezultat izvršenja embedded shell-a se prenosi u shell roditelj kao deo linije koju analizira shell. U prvom primeru, rezultat je bio string *tekst*. Otuda je shell roditelj analizirao liniju koja sadrži samo reč *tekst*, i pošto bi prva reč trebalo da predstavlja naziv komande, shell je potražio komandu tekst. Pošto je nije našao, prijavio je grešku.

Shell gneždenje (embedding)

```
ubuntu@ubuntu-VirtualBox:~$ echo $(cd /home/ubuntu; ls)
Desktop Documents Downloads examples.desktop Music Pictures Public Templates Videos
ubuntu@ubuntu-VirtualBox:~$
```

U ovom primeru smo izvršili **echo** rezultata embedded shell-a, a koji je izvršio pomeranje u direktorijum /home/ubuntu i potom listing sadržaja tog direktorijuma. Ali, bez obzira na pomeranje (cd) unutar command substitution, nakon izvršenja kompletne komande iz primera bi se i dalje ostalo u istom tekućem direktorijumu (koji je bio tekući i pre izvršenja same komande koja je data kao primer).

Bash istorija

- Bash čuva istoriju unesenih komandi
- Strelicama gore/dole se može kretati napred-nazad po istoriji komandi
- **!!** se koristi za izvršenje poslednje komande (tzv. *bang bang*)
- **!xx** se koristi za izvršenje poslednje komande koja počinje stringom xx, pri čemu umesto xx se pišu početni karakteri komande koja se želi ponoviti, a broj karaktera zavisi od toga koliko ih je potrebno da bi sigurno izvršili željenu komandu - izvršava se poslednja komanda koja počinje zadatim nizom karaktera
- Za prikaz poslednjih *n* komandi se koristi komanda **history n** gde se za *n* stavlja broj komandi koji želimo prikazati
- Za izvršenje komande pod brojem *n* iz istorije koristi se **!n**

Bash istorija - primeri

```
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
fajl1  fajl2
ubuntu@ubuntu-VirtualBox:~/Documents$ !!
ls
fajl1  fajl2
ubuntu@ubuntu-VirtualBox:~/Documents$ echo prvi
prvi
ubuntu@ubuntu-VirtualBox:~/Documents$ ls
fajl1  fajl2
ubuntu@ubuntu-VirtualBox:~/Documents$ !e
echo prvi
prvi
ubuntu@ubuntu-VirtualBox:~/Documents$ history 3
  782  ls
  783  echo prvi
  784  history 3
ubuntu@ubuntu-VirtualBox:~/Documents$ !783
echo prvi
prvi
ubuntu@ubuntu-VirtualBox:~/Documents$
```

U slučaju ponovnog izvršenja komande nekom od varijanata koje počinju !, ispisuje se i sama komanda u terminalu.

Bash istorija

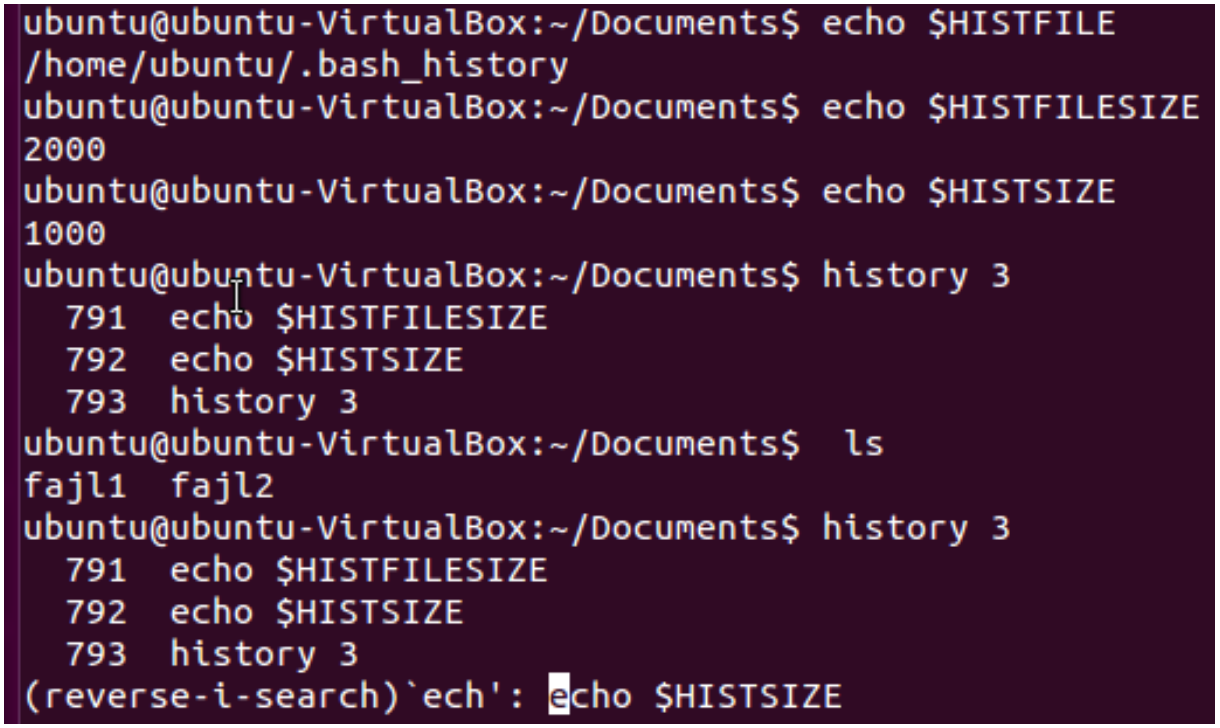
- Sama istorija se čuva u fajlu - koji fajl je u pitanju se može videti iz promenljive $HISTFILE
- Broj poslednjih komandi koji se čuva u ovom fajlu se može videti u promenljivoj $HISTFILESIZE
- Broj poslednjih komandi koji se čuva u trenutnom okruženju se može videti u promenljivoj $HISTSIZE
- Vrednosti navedenih promenljivih se mogu promeniti (na primer, HISTSIZE=10)
- CTRL+R omogućava pretragu za komandom
- Kucanje jednog space karaktera na početku komande sprečava da se ova komanda zapamti u istoriji

```
ubuntu@ubuntu-VirtualBox:~/Documents$ HISTSIZE=500
ubuntu@ubuntu-VirtualBox:~/Documents$ echo $HISTSIZE
500
ubuntu@ubuntu-VirtualBox:~/Documents$ HISTSIZE=1000
ubuntu@ubuntu-VirtualBox:~/Documents$ echo $HISTSIZE
1000
ubuntu@ubuntu-VirtualBox:~/Documents$
```

Primer promene vrednosti HISTSIZE promenljive

Za trajnu promenu treba editovati odgovarajući fajl (npr. .bashrc u Ubuntu)

Bash istorija - primeri



Ispred **ls** je stavljen space karakter pa ova komanda nije upisana u istoriju. U drugom ispisu istorije poslednje 3 komande, history 3 komanda se pojavila samo jednom iako je dva puta izvršena. Ako je komanda koja se izvrši identična poslednje izvršenoj komandi onda se ona ne unosi u istoriju.

CTRL+R daje mogućnost pretrage prikazane u poslednjoj liniji primera. Kucanjem termina pretrage, istorija izbacuje poslednju komandu koja odgovara zadatom terminu.