

ELEKTROTEHNIČKI FAKULTET UNIVERZITETA U BEOGRADU



PISANJE SKRIPTI U LINUKSU

– Diplomski rad –

Kandidat:

Darko Nitković 2010/413

Mentor:

doc. dr Zoran Čića

Beograd, Novembar 2016.

SADRŽAJ

SADRŽAJ.....	2
1. UVOD	4
2. LJUSKA	5
2.1. PRISTUP LJUSCI.....	5
3. RAD SA DATOTEKAMA I DIREKTORIJUMIMA	7
3.1. HIJERARHIJA DIREKTORIJUMA	7
3.2. PWD	7
3.3. APSOLUTNE I RELATIVNE PUTANJE.....	7
3.4. LS	8
3.5. FILE	9
3.6. LESS.....	9
3.7. CP	10
3.8. MV	10
3.9. RM	11
3.10. CAT.....	11
3.11. REDIREKCIJA STANDARDNOG IZLAZA I STANDARDNE GREŠKE	12
3.12. PROTOČNA OBRADA	13
3.13. UNIQ I SORT	13
3.14. WC	14
3.15. TEE	14
3.16. ECHO	14
3.17. DŽOKER ZNAKOVI	15
3.18. SPECIJALNI ZNAKOVI.....	16
3.19. JEDNOSTRUKI NAVODNICI	17
3.20. DVOSTRUKI NAVODNICI	18
3.21. OBRNUTI NAVODNICI.....	18
3.22. GREP	18
4. PISANJE KOMANDNIH SKRIPTOVA	21
4.1. HELLO_WORLD	21
4.2. PROMJENLJIVE.....	23
4.3. READ	25
4.4. IZLAZNI STATUS	25
4.5. LOGIČKI KONSTRUKTI && I 	26
4.6. IF NAREDBA	27
4.7. ELSE	27
4.8. ELIF	28
4.9. TEST	29
4.9.1. <i>Ispitivanje cjelobrojnih promjenljivih.</i>	29
4.9.2. <i>Rad sa stringovima</i>	30
4.9.3. <i>Ispitivanje datoteka</i>	32
4.10. KONSTRUKCIJA [[]].	34
4.11. CASE NAREDBA	36
4.12. WHILE	39
4.13. FOR.....	40
4.14. UNTIL	41
4.15. BREAK I CONTINUE	42

4.16.	FUNKCIJE.....	44
4.16.1.	<i>Definisanje i pozivanje funkcija</i>	44
4.16.2.	<i>Pozicioni parametri</i>	45
4.16.3.	<i>Lokalne promjenljive</i>	48
5.	ZAKLJUČAK.....	50
	LITERATURA.....	51

1.UVOD

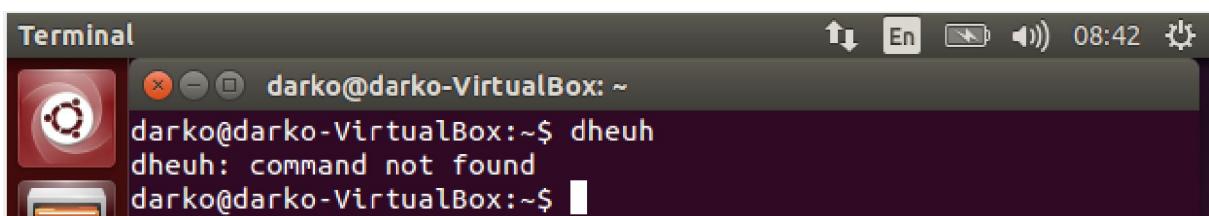
Cilj ovog rada je upoznavanje sa pisanjem komandnih skriptova u Linuks okruženju. Kako komandna linija podrazumijeva intenzivan rad sa datotekama i direktorijumima ova tehnika predstavlja veoma korisno sredstvo za automatizaciju posla. Dato je više konkretnih primjera koji treba da pomognu čitaocu da razumije skriptove i rad sa njima. Skriptovi realizovani u okviru ove teze biće priloženi u elektronskom obliku. Sam rad je organizovan u pet poglavlja.

Prvo poglavlje predstavlja uvod, i u njemu je dat kratak pregled predmeta samog rada, i kratko upoznavanje sa sadržajem ostalih poglavlja. Drugo poglavlje prezentuje lјusku *bash*, kao sredstvo interakcije između korisnika i operativnog sistema. U ovom poglavlju je predstavljen i emulator terminala kojim se pristupa lјusci. Treće poglavlje daje uvid u rad sa datotekama i direktorijumima. Objasnjena je hijerarhija, kao i osnovne komande za rad sa njima. U četvrtom poglavlju je opisano kreiranje komandnih skriptova, uz korišćenje komandi opisanih u trećem poglavlju. Peto poglavlje predstavlja zaključak u kom je dat pregled pozitivnih i negativnih strana ove tehnike.

2. LJUSKA

2.1. Pristup ljudstvu

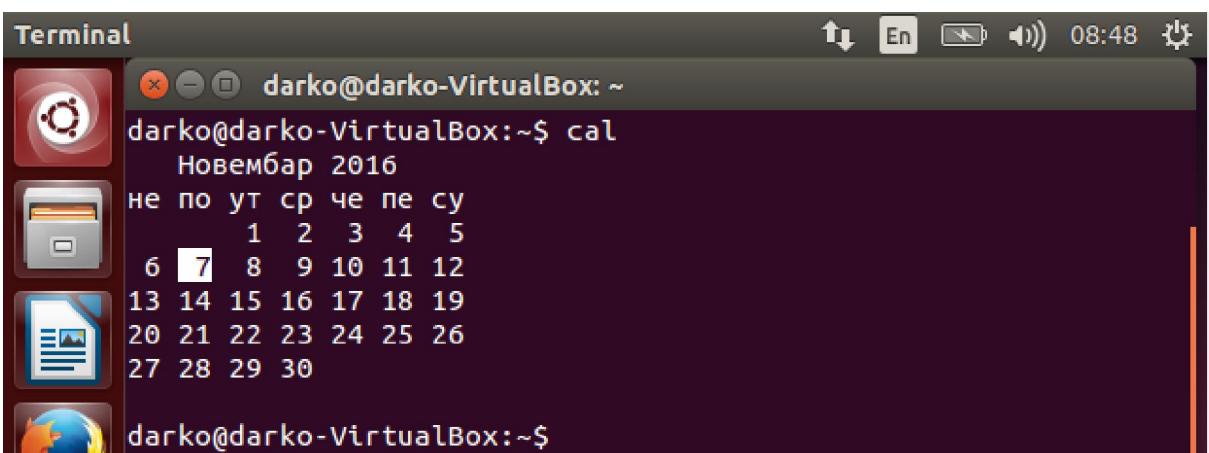
Ljuska je program – interpreter komandne linije – koji preuzima komande sa tastature i prosleđuje ih operativnom sistemu na izvršavanje [1]. Ljuska koja je prisutna u većini Linuks distribucija je *bash*. *Bash* je skraćenica imena *Bourne Again Shell* i predstavlja unaprijeđenu verziju originalne Unixove ljudske *Bourne Shell* čiji je autor Stiv Burn. Za pristup ljudstvu iz grafičkog okruženja koristi se program koji se zove emulator terminala. Nakon pokretanja emulatora terminala dostupan je odzivnik ljuske koji se pojavljuje kada god je ljuska spremna da prihvati unos. U zavisnosti od distribucije operativnog sistema format odzivnika može da varira ali u najvećem broju slučajeva sadrži: korisničko ime, ime računara na kom je operativni sistem instaliran, tekući radni direktorijum korisnika i znak za dolar. Komande koje se unose počinju iza znaka za dolar i na taj način se razdvajaju od odzivnika. Unosom neke nepostojeće komande ljuska izdaje upozorenje i pruža mogućnost za ponovni unos.



A screenshot of a Linux terminal window titled "Terminal". The window shows a dark-themed interface with a title bar containing the terminal name and the user's session information: "darko@darko-VirtualBox: ~". The main area of the terminal displays the following text:
dheuh
dheuh: command not found
darko@darko-VirtualBox:~\$

Slika 2.1.1. - Primjer unosa nepostojeće komande

Komanda *cal* prikazuje kalendar za tekući mjesec.



A screenshot of a Linux terminal window titled "Terminal". The window shows a dark-themed interface with a title bar containing the terminal name and the user's session information: "darko@darko-VirtualBox: ~". The main area of the terminal displays the following text:
Новембар 2016
не по ут ср че не су
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
darko@darko-VirtualBox:~\$

Slika 2.1.2. - Rezultat komande cal

Ljuska pruža mogućnost kretanja kroz istoriju unijetih komandi. Pritisom na taster strelica gore moguće je vidjeti prethodno unijetu komandu. Pritisom na taster strelica dole prethodno

unijeta komanda nestaje. Na ovaj način moguće je pregledati i do 500 prethodno unijetih komandi što zavisi od odgovarajućih podešavanja u operativnom sistemu. Tekuća sesija završava se zatvaranjem prozora klikom na x ili unosom komande exit.

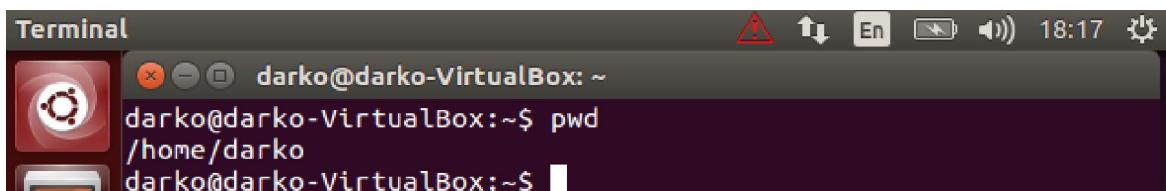
3. RAD SA DATOTEKAMA I DIREKTORIJUMIMA

3.1. Hjerarhija direktorijuma

Direktorijumi u Linuksu su organizovani u hjerarhijsku strukturu. Ovo podrazumijeva organizaciju direktorijuma u obliku stabla. Direktorijum koji je na samom vrhu ove strukture je korjenski. On se označava sa /. Unutar njega se nalaze direktorijumi u kojima su smješteni druge datoteke i direktorijumi. Korisnik u jednom trenutku može da se nalazi samo u jednom direktorijumu. Direktorijum u kome se korisnik trenutno nalazi naziva se tekući radni direktorijum. Direktorijum neposredno iznad je roditeljski direktorijum tekućem radnom direktorijumu.

3.2. pwd

Za prikazivanje tekućeg radnog direktorijuma koristi se komanda pwd.

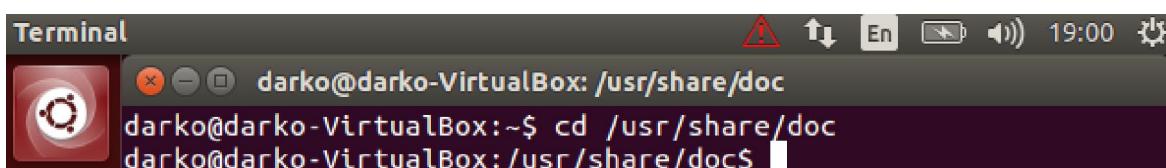


```
Terminal
darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ pwd
/home/darko
darko@darko-VirtualBox:~$
```

Slika 3.2.1. - Prikazivanje tekućeg radnog direktorijuma

3.3. Apsolutne i relativne putanje

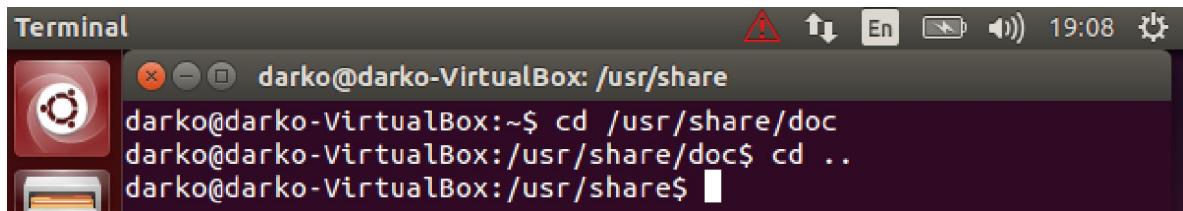
Referencirati neku datoteku ili direktorijum znači zadati putanju. Postoje dvije vrste putanja. To su absolutne i relativne putanje. Kada putanja počinje znakom /, to je puna ili absolutna putanja [2]. Putanja koja ne počinje ovim znakom je relativna. Za promjenu tekućeg radnog direktorijuma koristi se komanda cd. Postavljanje direktorijuma /usr/share/doc za tekući radni prikazano je na slici 3.3.1.. Ovim postupkom je iskorišćeno zadavanje absolutne putanje. Kreće se od korjenskog direktorijuma. Unutar njega se nalazi folder usr. Unutar ovog foldera nalazi se folder share. Unutar foldera share je folder doc u kome se nalaze razne dokumentacione datoteke.



```
Terminal
darko@darko-VirtualBox: /usr/share/doc
darko@darko-VirtualBox:~$ cd /usr/share/doc
darko@darko-VirtualBox:/usr/share/doc$
```

Slika 3.3.1. - Zadavanje absolutne putanje

Relativne putanje koriste znakove . i .. . Znak . predstavlja tekući radni direktorijum. Znak .. predstavlja roditeljski direktorijum tekućeg radnog direktorijuma. Za prelazak u roditeljski direktorijum tekućeg radnog direktorijuma koristi se naredba cd .. . Za prelazak iz direktorijuma /usr/share u /usr/share/doc unosi se cd doc jer je /usr/share/doc poddirektorijum direktorijuma /usr/share, u tom slučaju je ovo moguće, jer se radi o poddirektorijumu tekućeg radnog.

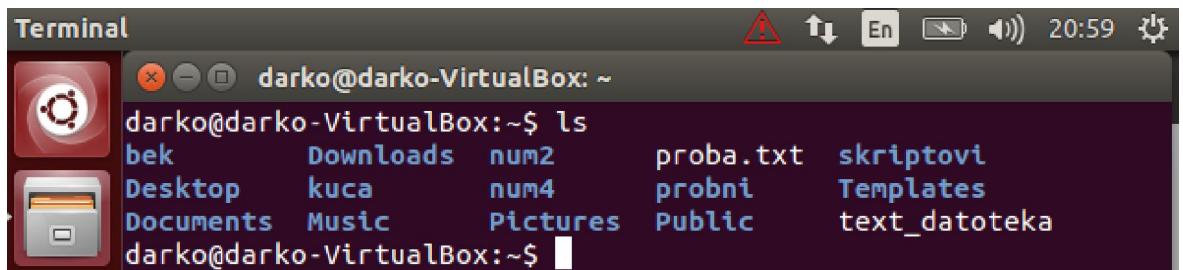


```
Terminal darko@darko-VirtualBox: /usr/share
darko@darko-VirtualBox:~$ cd /usr/share/doc
darko@darko-VirtualBox:/usr/share/doc$ cd ..
darko@darko-VirtualBox:/usr/share$
```

Slika 3.3.2. - Primjer upotrebe relativne putanje

3.4. ls

Ova komanda u svom osnovnom obliku služi za izlistavanje direktorijuma. Nije nužno da komanda ls izlistava sadržaj trenutnog direktorijuma. Komanda se može koristiti za izlistavanje bilo kog direktorijuma. Prikazani rezultat moguće je modifikovati dodavanjem opcija. Upotrebom ls -F /etc dodaje se indikatorski znak pored svakog imena (npr. / za direktorijum) . Konstrukcija ls -a /dev izlistava sve datoteke direktorijuma /dev uključujući i skrivene koje počinju tačkom.



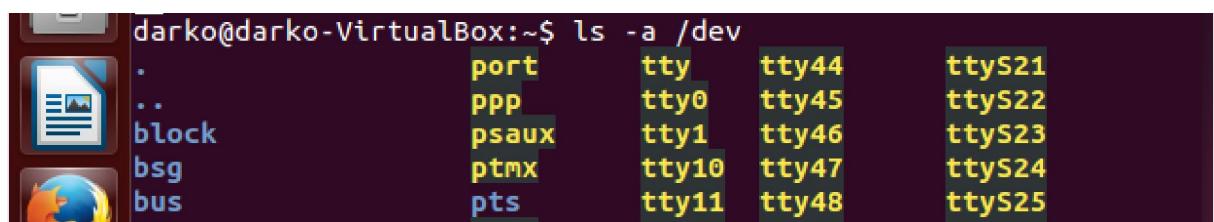
```
Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ ls
bek      Downloads  num2      proba.txt  skriptovi
Desktop   kuca     num4      probni    Templates
Documents Music    Pictures  Public    text_datoteka
darko@darko-VirtualBox:~$
```

Slika 3.4.1. - Izlistavanje tekućeg radnog direktorijuma



```
darko@darko-VirtualBox:~$ ls -F /etc
acpi/               magic
adduser.conf        magic.mime
alternatives/       mailcap
anacrontab         mailcap.order
apg.conf           manpath.config
apm/               mime.types
apparmor/          mke2fs.conf
apparmor.d/         modprobe.d/
apport/             modules
```

Slika 3.4.2. – Izlistavanje direktorijuma /etc uz dodavanje indikatorskog znaka



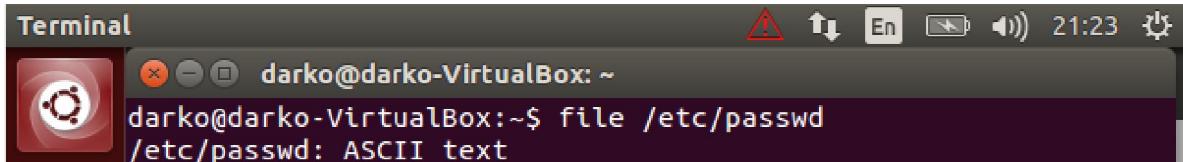
```
darko@darko-VirtualBox:~$ ls -a /dev
.                  port      tty      tty44     ttyS21
..                 ppp       tty0     tty45     ttyS22
block              psaux    tty1     tty46     ttyS23
bsg                ptmx    tty10    tty47     ttyS24
bus                pts     tty11    tty48     ttyS25

```

Slika 3.4.3. – Izlistavanje direktorijuma /dev uključujući i skrivene datoteke

3.5. file

Komanda file se koristi za određivanje tipa datoteke. Unošenjem file *datoteka* određuje se tip datoteke *datoteka*. Rezultat komande file /etc/passwd pokazuje da je datoteka u kojoj se čuva lista korisničkih naloga tekstualnog tipa.

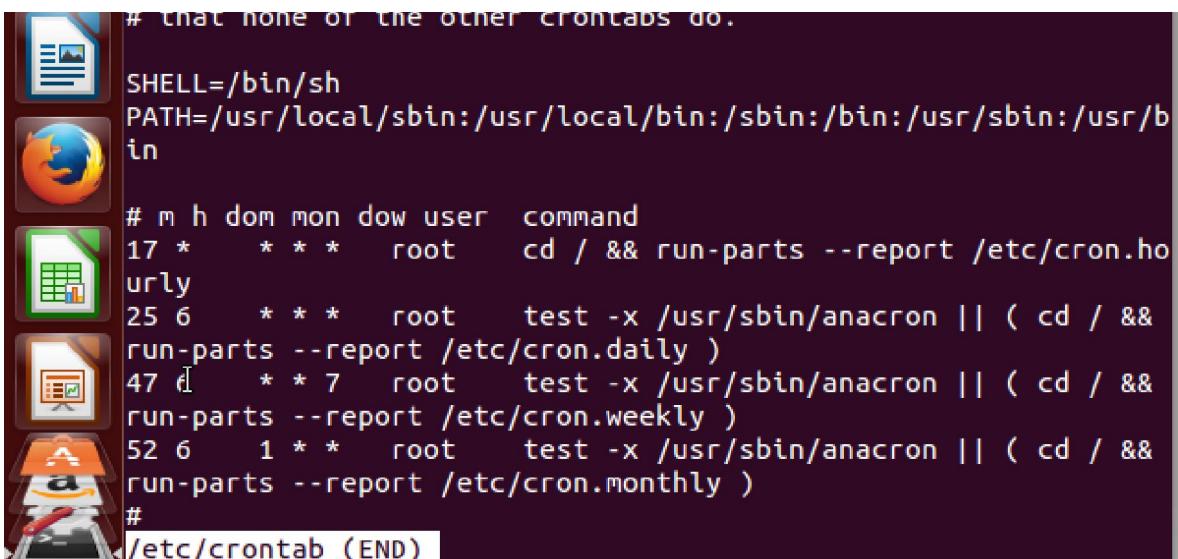


```
Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ file /etc/passwd
/etc/passwd: ASCII text
```

Slika 3.5.1. – Određivanje tipa datoteke /etc/passwd

3.6. less

Komanda se koristi za pregledanje sadržaja tekstualnih datoteka. Formata je: less *datoteka*. Mnoge konfiguracione datoteke su tekstualnog tipa. Skriptovi su takođe datoteke tekstualnog tipa. Zbog toga je ova komanda korisna jer pruža uvid u sadržaj navedenih datoteka. Naredbom less /etc/crontab prikazuje se sadržaj datoteke u kojoj je definisano kada će se izvršavati automatski poslovi.



```
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/b
in

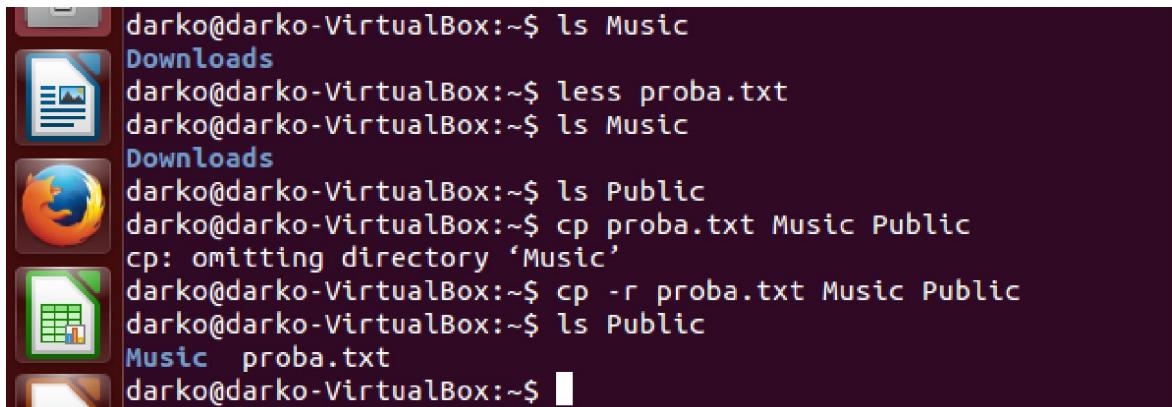
# m h dom mon dow user  command
17 *      * * *    root    cd / && run-parts --report /etc/cron.ho
urly
25 6      * * *    root    test -x /usr/sbin/anacron || ( cd / &&
run-parts --report /etc/cron.daily )
47 1      * * 7    root    test -x /usr/sbin/anacron || ( cd / &&
run-parts --report /etc/cron.weekly )
52 6      1 * *    root    test -x /usr/sbin/anacron || ( cd / &&
run-parts --report /etc/cron.monthly )
#
/etc/crontab (END)
```

Slika 3.6.1. – Primjer upotrebe komande less

Moguće je kretanje kroz prikazane rezultate ako postoji više stranica. Taster page up vrši pomjeranje za jednu stranu unazad. Taster page down vrši pomjeranje za jednu stranu unaprijed. Strelica na gore vrši pomjeranje za jedan red nagore. Strelica nadole vrši pomjeranje za jedan red nadole. Taster G pomjera na kraj tekstualne datoteke. Izlazak iz rezultata prikazanih na ekranu vrši se pritiskom tastera Q.

3.7. cp

U svom osnovnom obliku vrši kopiranje datoteka. Za kopiranje datoteke1 u datoteku2 potrebno je unijeti sledeće : cp datoteka1 datoteka2. Ovom naredbom moguće je kopirati više stavki u jedan direktorijum. Unošenjem cp -r proba.txt Music Public vršimo kopiranje datoteke proba.txt i direktorijuma Music u prazan direktorijum Public. Kada se vrši kopiranje direktorijuma potrebno je zadati opciju -r jer time vršimo rekursivno kopiranje. To znači da se vrši i kopiranje sadržaja datog direktorijuma.

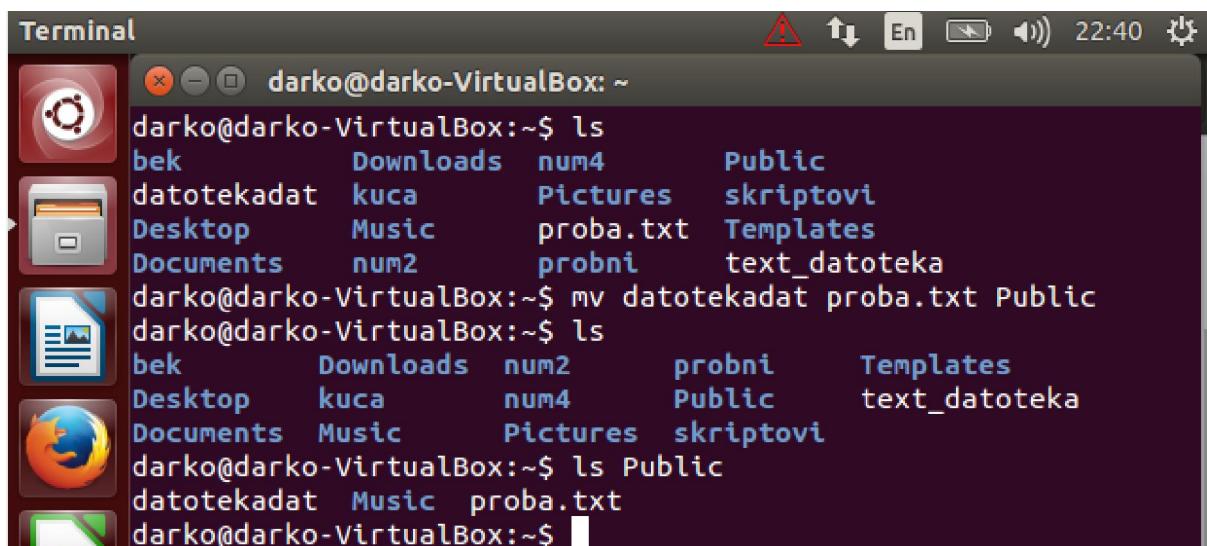


```
darko@darko-VirtualBox:~$ ls Music
Downloads
darko@darko-VirtualBox:~$ less proba.txt
darko@darko-VirtualBox:~$ ls Music
Downloads
darko@darko-VirtualBox:~$ ls Public
darko@darko-VirtualBox:~$ cp proba.txt Music Public
cp: omitting directory 'Music'
darko@darko-VirtualBox:~$ cp -r proba.txt Music Public
darko@darko-VirtualBox:~$ ls Public
Music proba.txt
darko@darko-VirtualBox:~$
```

Slika 3.7.1. – Primjer upotrebe komande cp

3.8. mv

Ova komanda omogućava preimenovanje ili premještanje datoteka u zavisnosti za šta se koristi. Unosom mv dat1 dat2 vrši se preimenovanje datoteke imena dat1 u datoteku dat2, ukoliko dat2 ne postoji. Može se koristiti i za premještanje više datoteka u direktorijum. Ako se žele premjestiti datoteke datotekadat i proba.txt u direktorijum Public onda se to vrši unosom naredbe: mv datotekadat proba.txt Public.

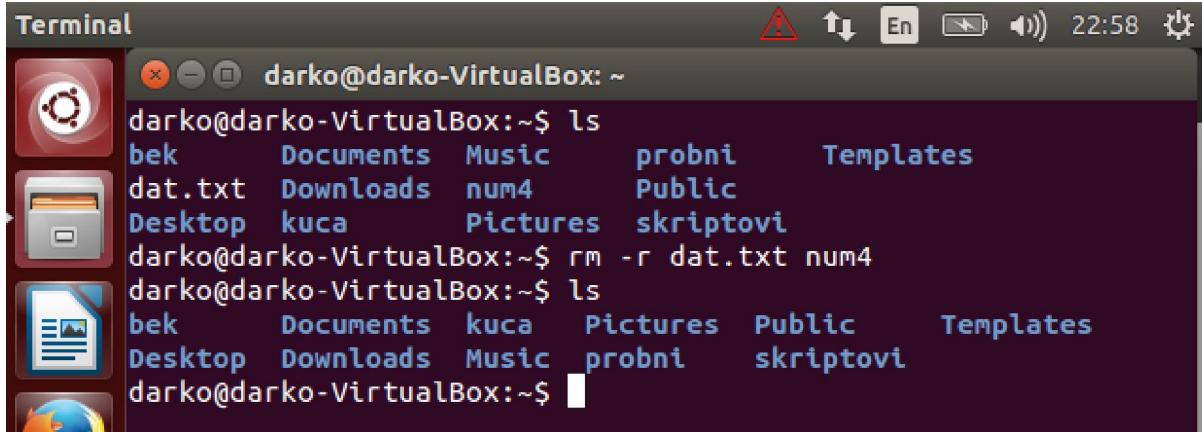


```
Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ ls
bek      Downloads  num4      Public
datotekadat  kuca    Pictures  skriptovi
Desktop    Music     proba.txt  Templates
Documents  num2      probni    text_datoteka
darko@darko-VirtualBox:~$ mv datotekadat proba.txt Public
darko@darko-VirtualBox:~$ ls
bek      Downloads  num2      probni    Templates
Desktop  kuca      num4      Public    text_datoteka
Documents Music     Pictures  skriptovi
darko@darko-VirtualBox:~$ ls Public
datotekadat  Music  proba.txt
darko@darko-VirtualBox:~$
```

Slika 3.8.1. – Primjer upotrebe komande mv

3.9. rm

Naredba rm služi za uklanjanje datoteka i direktorijuma. Uklanjanje datoteke dat.txt i direktorijuma num2 vrši se unošenjem rm -r dat.txt num4. Kada se uklanja direktorijum potrebno je unijeti -r. U suprotnom, uklanjanje neće biti uspješno.

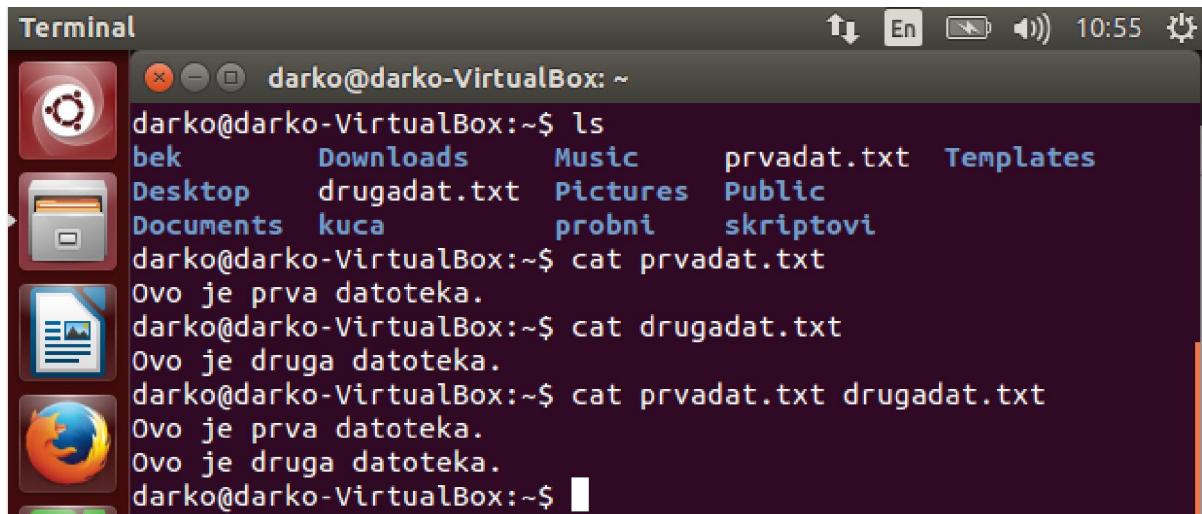


```
darko@darko-VirtualBox:~$ ls
bek      Documents  Music      probni    Templates
dat.txt  Downloads  num4      Public
Desktop  kuca      Pictures   skriptovi
darko@darko-VirtualBox:~$ rm -r dat.txt num4
darko@darko-VirtualBox:~$ ls
bek      Documents  kuca      Pictures  Public    Templates
Desktop  Downloads  Music     probni   skriptovi
darko@darko-VirtualBox:~$
```

Slika 3.9.1. – Primjer upotrebe komande rm

3.10. cat

Vrši čitanje jedne ili više datoteka pa rezultate prikazuje na ekran.

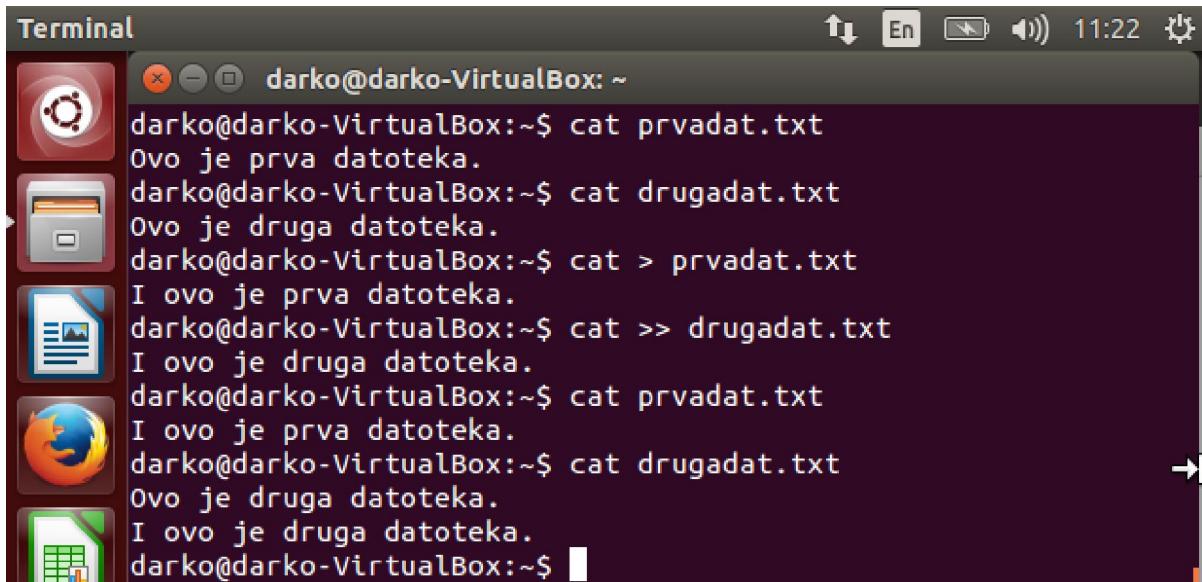


```
darko@darko-VirtualBox:~$ ls
bek      Downloads  Music      prvadat.txt  Templates
Desktop  drugadat.txt  Pictures  Public
Documents  kuca      probni    skriptovi
darko@darko-VirtualBox:~$ cat prvadat.txt
Ovo je prva datoteka.
darko@darko-VirtualBox:~$ cat drugadat.txt
Ovo je druga datoteka.
darko@darko-VirtualBox:~$ cat prvadat.txt drugadat.txt
Ovo je prva datoteka.
Ovo je druga datoteka.
darko@darko-VirtualBox:~$
```

Slika 3.10.1. – Primjer upotrebe komande cat

Ako se ne navedu argumenti komande cat i stisne ENTER cat čeka korisnikov unos sa tastature. Nakon što korisnik završi sa unosom i pritisne CTRL+D cat taj unos prikazuje na ekranu. Cat se takođe koristi se operatorima redirekcije > i >>. Za konstrukciju cat > prvadat.txt cat vrši preusmjeravanje. Korisnik unosi željeni tekst i pritiska CTRL+D. Cat ovaj unos ne prikazuje na ekranu već ga preusmjerava u prvadat.txt. Upotreboom operatora redirekcije > preko sadržaja prvadat.txt upisuje se unijeti tekst. U tom slučaju gubi se prvobitni sadržaj datoteke prvadat.txt.

Operator redirekcije >> u kombinaciji sa cat vrši dodavanje na kraj datoteke. Naredba cat >> drugadat.txt neće prebrisati sadržaj drugadat.txt. Unijeti tekst će dodati na kraj ove datoteke.

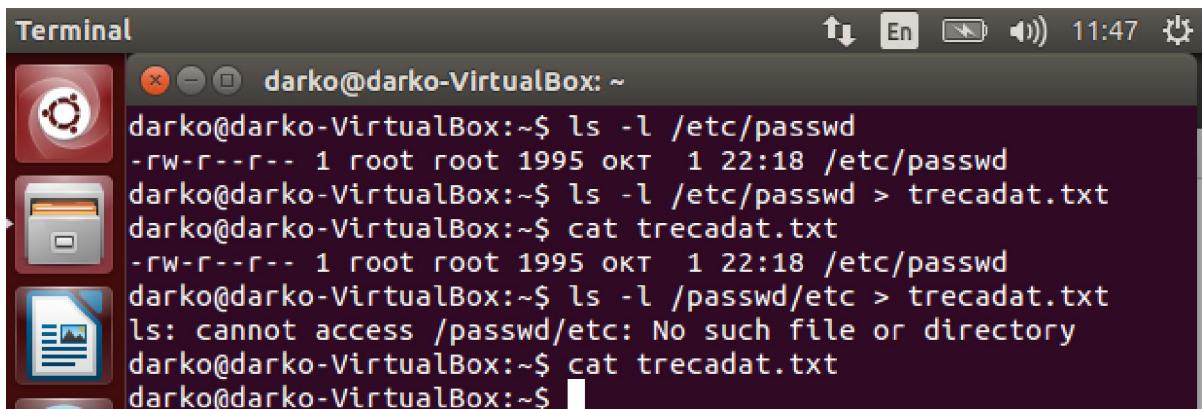


```
Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ cat prvadat.txt
Ovo je prva datoteka.
darko@darko-VirtualBox:~$ cat drugadat.txt
Ovo je druga datoteka.
darko@darko-VirtualBox:~$ cat > prvadat.txt
I ovo je prva datoteka.
darko@darko-VirtualBox:~$ cat >> drugadat.txt
I ovo je druga datoteka.
darko@darko-VirtualBox:~$ cat prvadat.txt
I ovo je prva datoteka.
darko@darko-VirtualBox:~$ cat drugadat.txt
Ovo je druga datoteka.
I ovo je druga datoteka.
darko@darko-VirtualBox:~$
```

Slika 3.10.2. – Upotreba komande cat sa operatorima redirekcije

3.11. Redirekcija standardnog izlaza i standardne greške

Unosom ls -l /etc/passwd > trecadat.txt vrši se preusmjeravanje izlaza komande ls u trecadat.txt. Ako se sada unese ls -l /passwd/etc > trecadat.txt dobiće se poruka o grešci. Ova greška je prikazana na ekranu jer direktorijum /passwd/etc ne postoji. Operatorom > je zadato da se u trecadat.txt preusmjeri izlaz komande ls a ne greška. Datoteka trecadat.txt je sada prazna. Operator > vrši upisivanje ispočetka. Kako je došlo do greške upisivanje nije uspjelo. Ishod je prazna datoteka trecadat.txt.

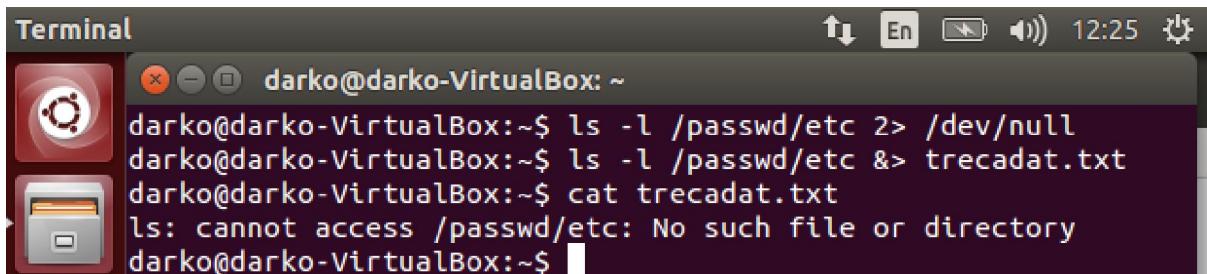


```
Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ ls -l /etc/passwd
-rw-r--r-- 1 root root 1995 окт 1 22:18 /etc/passwd
darko@darko-VirtualBox:~$ ls -l /etc/passwd > trecadat.txt
darko@darko-VirtualBox:~$ cat trecadat.txt
-rw-r--r-- 1 root root 1995 окт 1 22:18 /etc/passwd
darko@darko-VirtualBox:~$ ls -l /passwd/etc > trecadat.txt
ls: cannot access /passwd/etc: No such file or directory
darko@darko-VirtualBox:~$ cat trecadat.txt
darko@darko-VirtualBox:~$
```

Slika 3.11.1. – Ilustracija preusmjeravanja standardnog izlaza

Za redirekciju greške koristi se deskriptor datoteke. Sastoji se od standardnog ulaza, izlaza i greške. Označavaju se brojevima 0, 1, 2. Broj 1 se ne piše jer se podrazumijeva. Preusmjeravanje je moguće izvršiti u tzv "kofu za bitove." To je sistemska datoteka /dev/null. Ono što se tamo

preusmjeri ne može više biti prikazano. Npr. ls -l /passwd/etc 2> /dev/null preusmjerava grešku u /dev/null. Direktorijum /passwd/etc ne postoji. Greška ovom notacijom je tamo preusmjerena. Nije prikazana na ekranu. Naredba ls -l /passwd/etc &> trecadat.txt vrši preusmjeravanje i izlaza i greške ls -l /passwd/etc u trecadat.txt.

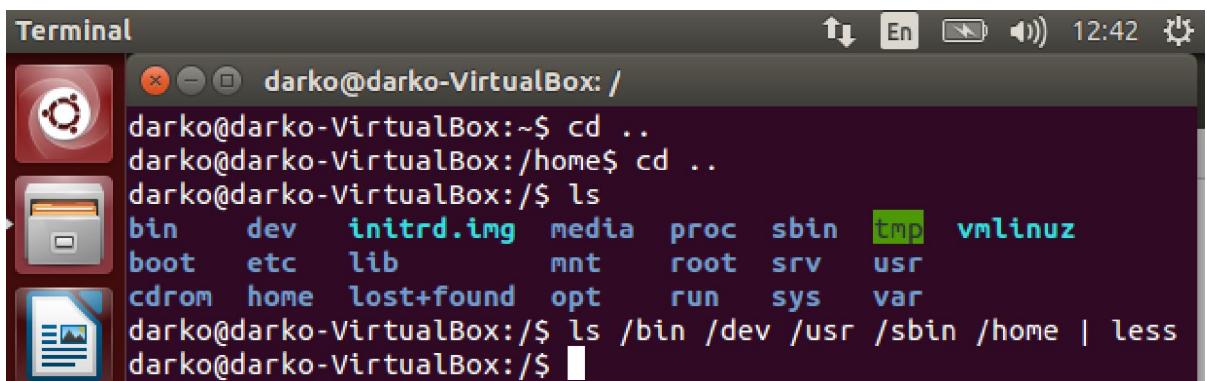


```
Terminal
darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ ls -l /passwd/etc 2> /dev/null
darko@darko-VirtualBox:~$ ls -l /passwd/etc &> trecadat.txt
darko@darko-VirtualBox:~$ cat trecadat.txt
ls: cannot access /passwd/etc: No such file or directory
darko@darko-VirtualBox:~$
```

Slika 3.11.2. – Preusmjeravanje greške različitim notacijama

3.12. Protočna obrada

Omogućava dovođenje izlaza jedne naredbe na ulaz druge naredbe. Koristi se sa operatorom |. Ima oblik: naredba1 | naredba2. Kao naredba2 može da se koristi bilo koja komanda koja prihvata izlaz naredbe1 kao svoj ulaz. Na slici 3.12.1. su izlistani neki direktorijumi korjenskog direktorijuma. Taj rezultat je proslijeđen komandi less.

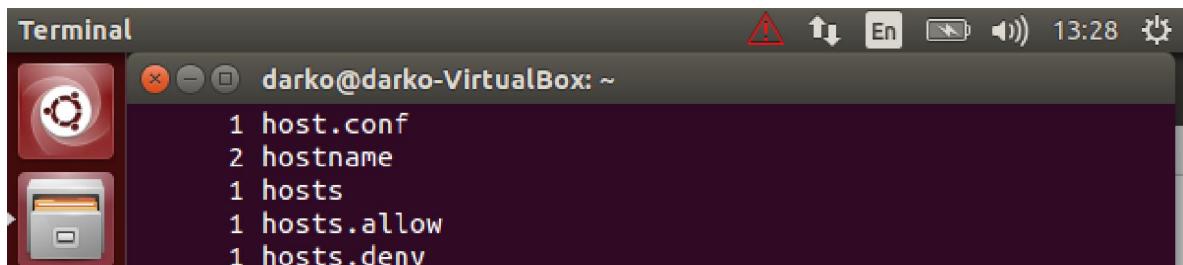


```
Terminal
darko@darko-VirtualBox: /
darko@darko-VirtualBox:~$ cd ..
darko@darko-VirtualBox:/home$ cd ..
darko@darko-VirtualBox:/$
darko@darko-VirtualBox:/$ ls
bin      dev      initrd.img   media   proc   sbin   tmp   vmlinuz
boot    etc      lib          mnt     root   srv   usr
cdrom   home    lost+found   opt     run    sys   var
darko@darko-VirtualBox:/$ ls /bin /dev /usr /sbin /home | less
darko@darko-VirtualBox:/$
```

Slika 3.12.1. – Primjer protočne obrade

3.13. uniq i sort

Protočnu obradu je moguće učiniti složenijom. To se postiže uvrštavanjem više naredbi u obradu. Naredba ls /bin /sbin /etc | sort | uniq -c | less je primjer ove primjene. Izlistava se sadržaj /bin /sbin /etc. Taj rezultat se sortira po abecednom redu komandom sort. Naredba uniq -c vrši numeraciju koliko se puta red ponavlja. Sve se to prosleđuje u less. Naredba less prikazuje rezultat na ekranu. Naredba uniq u svom osnovnom obliku eliminiše duplike redova. Ovu komandu je moguće modifikovati, kao i sve druge. Unosom man komanda prikazuje se detaljno uputstvo za komandu. Takođe, u uputstvu se prikazuje i spisak opcija za modifikaciju.

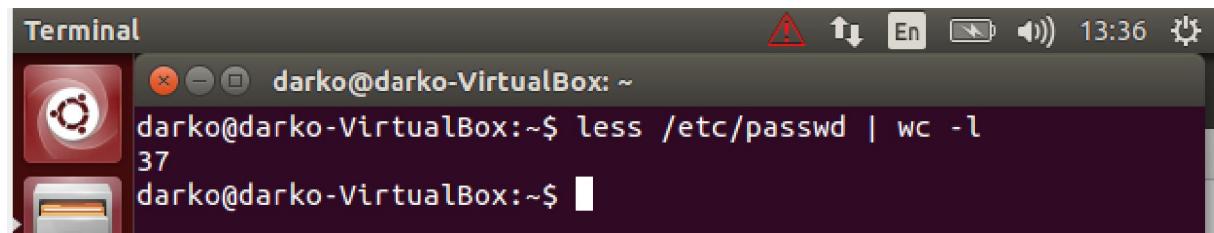


```
Terminal darko@darko-VirtualBox: ~
1 host.conf
2 hostname
1 hosts
1 hosts.allow
1 hosts.deny
```

Slika 3.13.1. – Efekat numerisanja redova pomoću uniq -c

3.14. wc

Prikazuje broj redova, riječi i bajtova u datotekama. Modifikacija wc -l prikazuje samo broj redova.

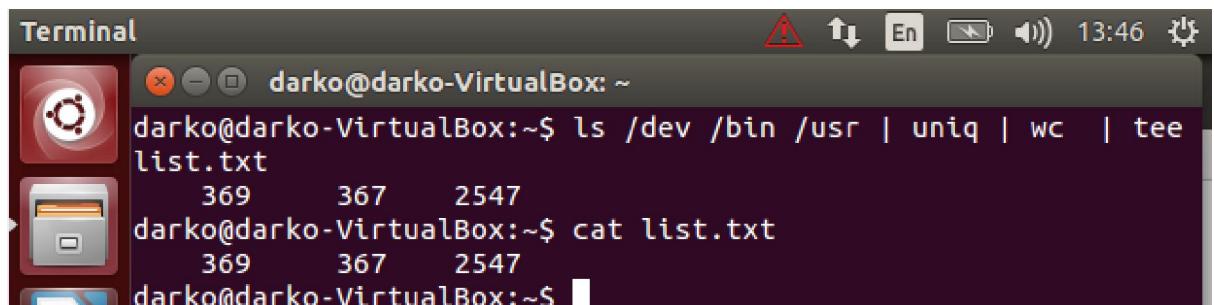


```
Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ less /etc/passwd | wc -l
37
darko@darko-VirtualBox:~$
```

Slika 3.14.1. – Prikazivanje broja redova u datoteci

3.15. tee

Nekada je potrebno međurezultate obrade sačuvati u tekstualnu datoteku. To se postiže ovom komandom.

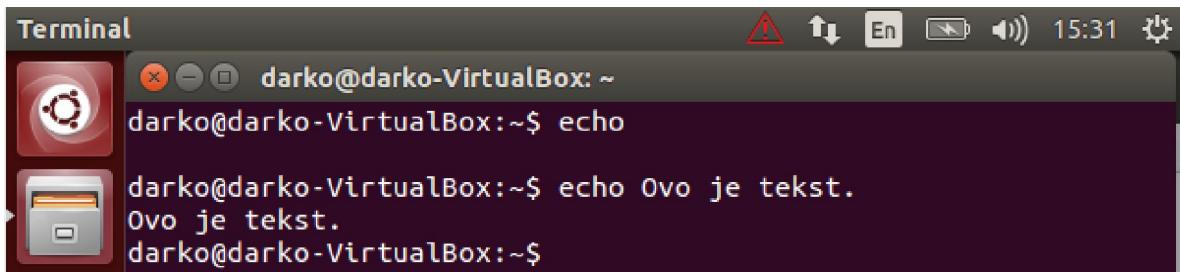


```
Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ ls /dev /bin /usr | uniq | wc | tee list.txt
369 367 2547
darko@darko-VirtualBox:~$ cat list.txt
369 367 2547
darko@darko-VirtualBox:~$
```

Slika 3.15.1. – Smještanje rezultata protočne obrade u tekstualnu datoteku

3.16. echo

Ispisuje parametre komandne linije na standardnom izlazu. Ako parametri nisu zadati ispisuje se prazan red.



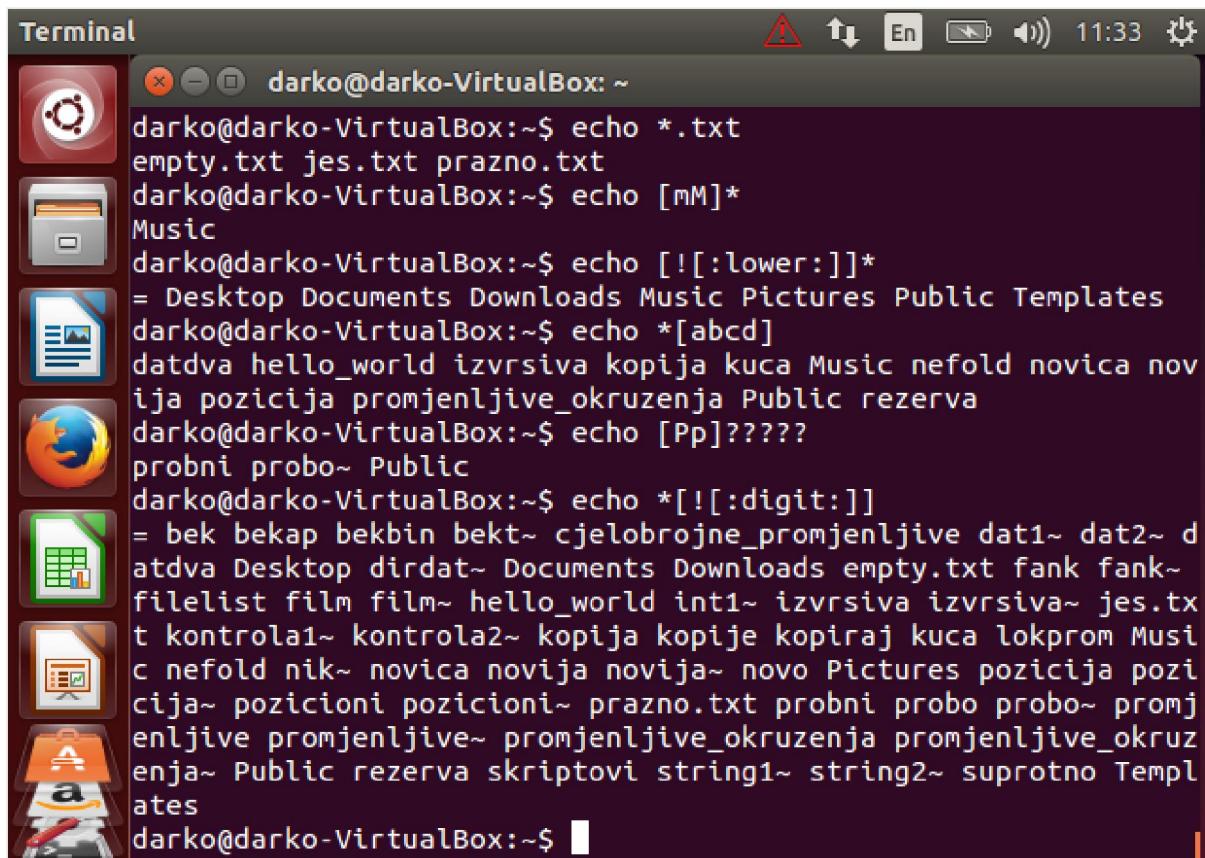
```
darko@darko-VirtualBox:~$ echo Ovo je tekst.  
Ovo je tekst.  
darko@darko-VirtualBox:~$
```

Slika 3.16.1. – Primjer upotrebe komande echo

3.17. Džoker znakovi

Komandna linija između ostalog podrazumijeva intenzivan rad sa datotekama. U tu svrhu često je potrebno zadati imena datoteka grupno. Takav postupak se naziva globiranje. U tu svrhu se koriste džoker znakovi. Neki od načina upotrebe džoker znakova su:

- *.txt Znak * vrši zamjenu bilo kojih znakova koji se nalaze prije .txt. Ovim se omogućava zadavanje imena svih datoteka koji se završavaju na .txt.
- [mM]* Zadaju se imena svih datoteka koje počinju ili sa m ili sa M.
- [![:lower:]]* Zadaju se sve datoteke koje ne počinju malim slovom.
- *[abcd] Sve datoteke koje se završavaju sa a, b , c, d.
- [Pp]????? Sve datoteke koje počinju ili sa velikim ili malim p iza kojeg sledi tačno pet znakova.
- *[![:digit:]] Sve datoteke koje se ne završavaju cifrom.



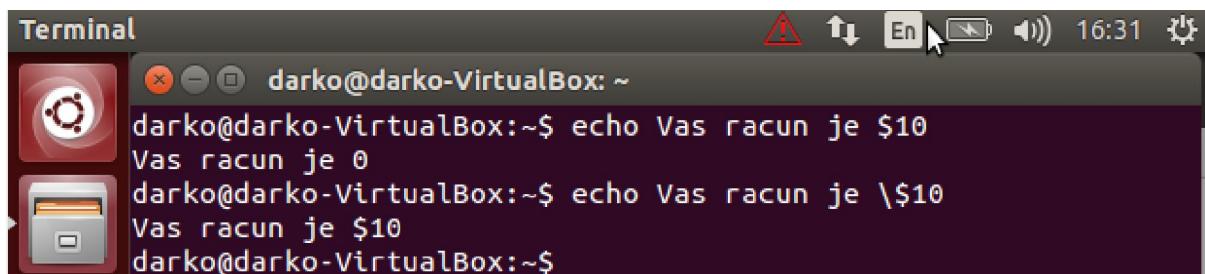
```

Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ echo *.txt
empty.txt jes.txt prazno.txt
darko@darko-VirtualBox:~$ echo [mM]*
Music
darko@darko-VirtualBox:~$ echo [![:lower:]]*
= Desktop Documents Downloads Music Pictures Public Templates
darko@darko-VirtualBox:~$ echo *[abcd]
datdva hello_world izvrsiva kopija kuca Music nefold novica novija pozicija promjenljive_okruzenja Public rezerva
darko@darko-VirtualBox:~$ echo [Pp]?????
probni probo~ Public
darko@darko-VirtualBox:~$ echo *[![:digit:]]
= bek bekap bekbin bekt~ cjelobrojne_promjenljive dat1~ dat2~ d
atdva Desktop dirdat~ Documents Downloads empty.txt fank fank~ filelist film film~ hello_world int1~ izvrsiva izvrsiva~ jes.tx
t kontrola1~ kontrola2~ kopija kopije kopiraj kuca lokprom Musi
c nefold nik~ novica novija novija~ novo Pictures pozicija pozicija~ pozicioni pozicioni~ prazno.txt probni probo probo~ promjenljive_promjenljive~ promjenljive_okruzenja promjenljive_okruzenja~ Public rezerva skriptovi string1~ string2~ suprotno Templa
tes
darko@darko-VirtualBox:~$
```

Slika 3.17.1. – Upotreba džoker znakova

3.18. Specijalni znakovi

Postoji grupa znakova koja pored svog osnovnog literalnog značenja ima i posebno značenje za ljudsku. To je takozvano metaliteralno značenje. U te znakove ubrajaju se: &, *, ?, [], < >, |, (), ', #, \$, ^, ', " , { } , ; , \. Znak \ eliminiše svojstvo svih specijalnih znakova. U prvom primjeru \\$1 se zamjenjuje sa praznim stringom. Znak za dolar se koristi za dohvatanje sadržaja u promjenljivoj. Kako promjenljiva nije definisana ta vrijednost se mijenja praznim stringom. U drugom primjeru specijalno značenje ovog znaka je uklonjeno pomoću \ . U tom slučaju znak za dolar ima svoje literalno značenje.



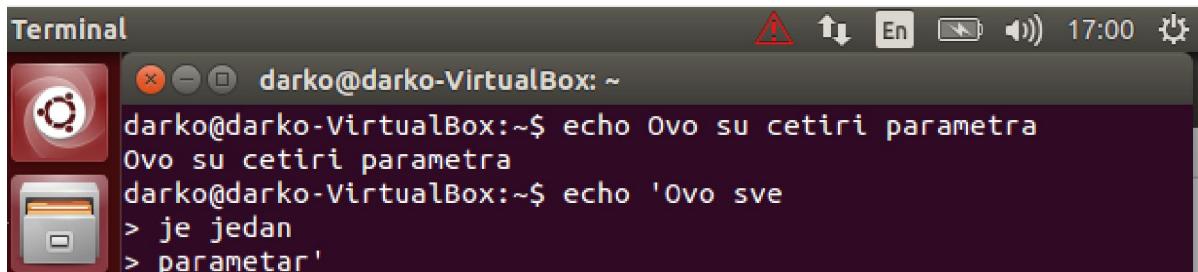
```

Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ echo Vas racun je $10
Vas racun je 0
darko@darko-VirtualBox:~$ echo Vas racun je \$10
Vas racun je $10
darko@darko-VirtualBox:~$
```

Slika 3.18.1. – Efekat specijalnog znaka \

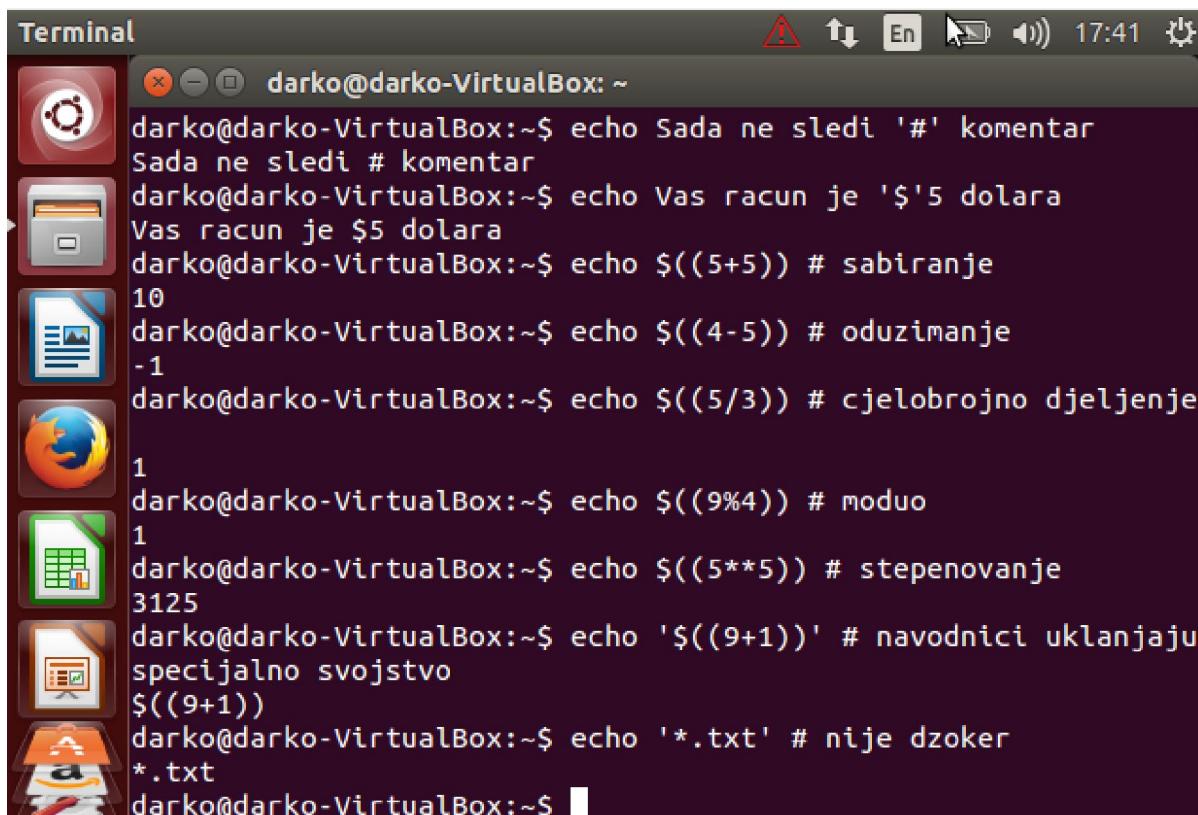
3.19. Jednostruki navodnici

Eliminišu specijalno značenje svih specijalnih znakova. Razmak omogućava da se razgraniči početak i kraj parametara. Ako postoji sedam parametara koji su okruženi jednostrukim navodnicima onda se oni tretiraju kao jedan parametar. Znak # predstavlja početak komentara. Sve što se iza njega nalazi ljudska ne čita. Moguće je vršiti određene aritmetičke operacije sa cijelim brojevima. Za ekspansiju aritmetičkih izraza koristi se sledeći oblik komande: \$((izraz)) gdje je izraz aritmetički izraz koji se sastoji od vrijednosti i aritmetičkih parametara [1].



```
darko@darko-VirtualBox:~$ echo Ovo su cetiri parametra
Ovo su cetiri parametra
darko@darko-VirtualBox:~$ echo 'Ovo sve
> je jedan
> parametar'
```

Slika 3.19.1. – Primjer upotrebe jednostrukih navodnika

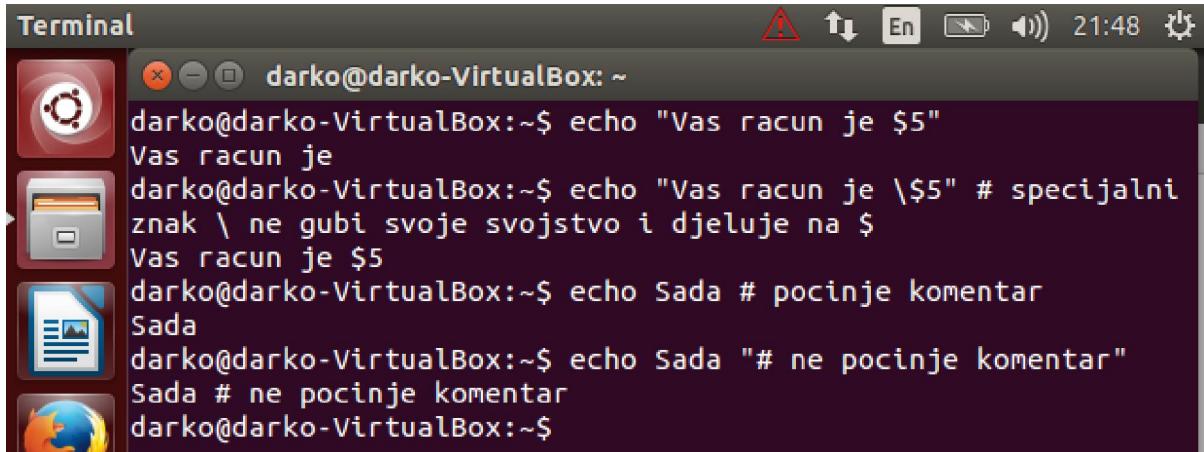


```
darko@darko-VirtualBox:~$ echo Sada ne sledi '#' komentar
Sada ne sledi # komentar
darko@darko-VirtualBox:~$ echo Vas racun je '$'5 dolara
Vas racun je $5 dolara
darko@darko-VirtualBox:~$ echo $((5+5)) # sabiranje
10
darko@darko-VirtualBox:~$ echo $((4-5)) # oduzimanje
-1
darko@darko-VirtualBox:~$ echo $((5/3)) # cjelobrojno djeljenje
1
darko@darko-VirtualBox:~$ echo $((9%4)) # moduo
1
darko@darko-VirtualBox:~$ echo $((5**5)) # stepenovanje
3125
darko@darko-VirtualBox:~$ echo '$((9+1))' # navodnici uklanjaju
specijalno svojstvo
$((9+1))
darko@darko-VirtualBox:~$ echo '*.txt' # nije dzoker
*.txt
darko@darko-VirtualBox:~$
```

Slika 3.19.2. – Efekat upotrebe jednostrukih navodnika

3.20. Dvostruki navodnici

Predstavljaju blaži kriterijum u odnosu na jednostrukе. To znači da određeni specijalni znakovi zadržavaju svoje značenje. To su: \$, ', \.



The screenshot shows a terminal window titled "Terminal" with the command line "darko@darko-VirtualBox:~\$". The user types several commands involving double quotes:

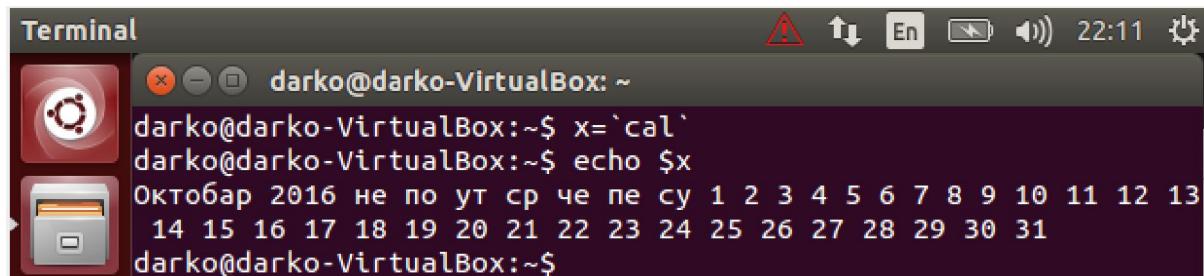
- "echo "Vas racun je \$5"
- "echo "Vas racun je \\$5" # specijalni znak \ ne gubi svoje svojstvo i djeluje na \$"
- "echo Sada # pocinje komentar Sada"
- "echo Sada "# ne pocinje komentar"

The output shows that the dollar sign (\$) and hash (#) are preserved as literal characters when enclosed in double quotes, demonstrating the double quote behavior.

Slika 3.20.1. – Primjer upotrebe dvostrukih navodnika

3.21. Obrnuti navodnici

Ne igraju ulogu kod specijalnih znakova. Osnovna njihova uloga je da se eliminišu novi redovi. Takođe sve višestruke razmake i tabulatore zamjenjuju jednostrukim. Takav izlaz se dodjeljuje nekoj promjenljivoj.



The screenshot shows a terminal window titled "Terminal" with the command line "darko@darko-VirtualBox:~\$". The user types the command "x=`cal`" and then "echo \$x". The output is:

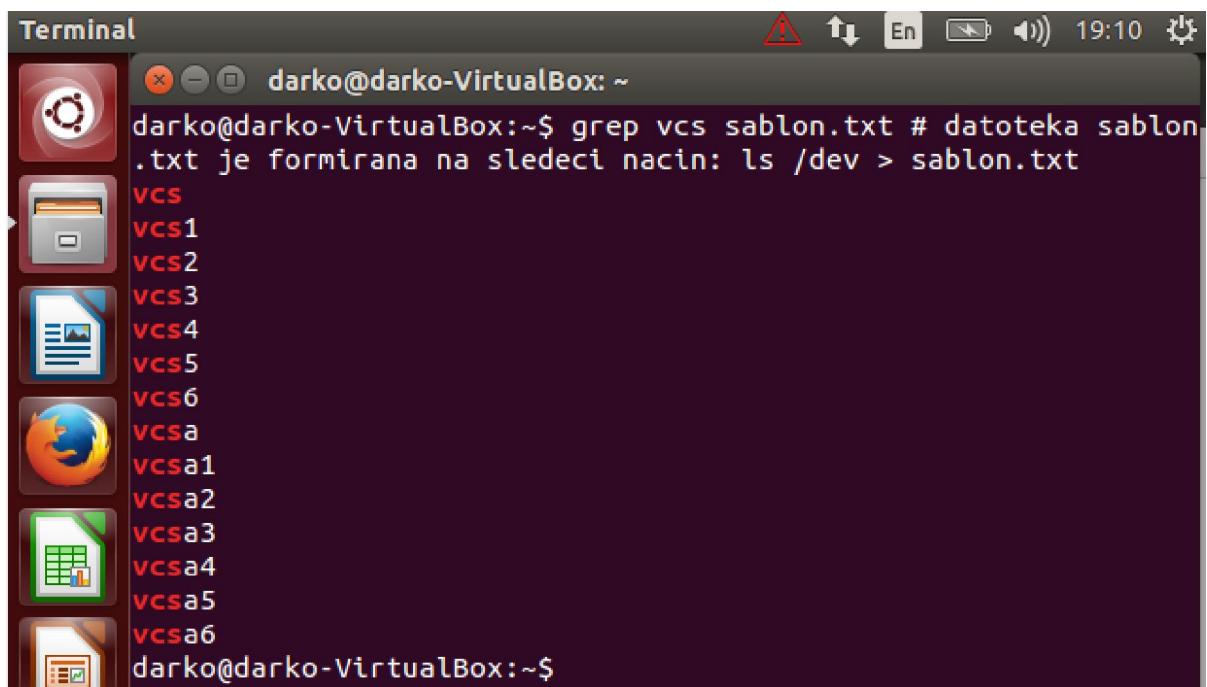
```
Октобар 2016
 НЕ ПО УТ СР ЧЕ НЕ СУ 1 2 3 4 5 6 7 8 9 10 11 12 13
 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
```

This demonstrates that inverted quotes (`) are used to store the output of a command into a variable, and then expanded when used in a subsequent echo command.

Slika 3.21.1. – Primjer upotrebe obrnutih navodnika

3.22. grep

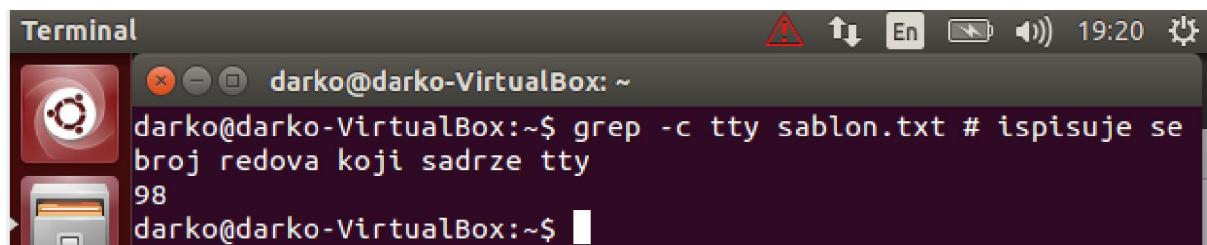
Vrši pretraživanje redova teksta sa svog ulaza koji sadrži određeni šablon. Taj ulaz može biti tastatura ili druga datoteka. Regularni izrazi predstavljaju složenije šablone. Ima oblik grep *šablon*. Unosom ovakve konstrukcije vrši se ispis redova teksta po *šablonu*, pri čemu je tekst koji se pretražuje unijet sa tastature. Unos grep vcs sablon.txt vrši ispis svih redova datoteke sablon.txt koji sadrže vcs. Konstrukcija grep -c tty sablon.txt ispisuje broj redova datoteke sablon.txt koji sadrže tty. Konstrukcija grep -v -i ovo *tdat.txt vrši ispis svih redova datoteke koje se završavaju na *tdat.txt i koji ne sadrže riječ ovo. Dodavanjem opcije -i zadaje se da se ne pravi razlika između malih i velikih slova.



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark purple background and contains the following text:

```
darko@darko-VirtualBox: ~$ grep vcs sablon.txt # datoteka sablon.txt je formirana na sledeci nacin: ls /dev > sablon.txt
vcs
vcs1
vcs2
vcs3
vcs4
vcs5
vcs6
vcsa
vcsa1
vcsa2
vcsa3
vcsa4
vcsa5
vcsa6
darko@darko-VirtualBox: ~$
```

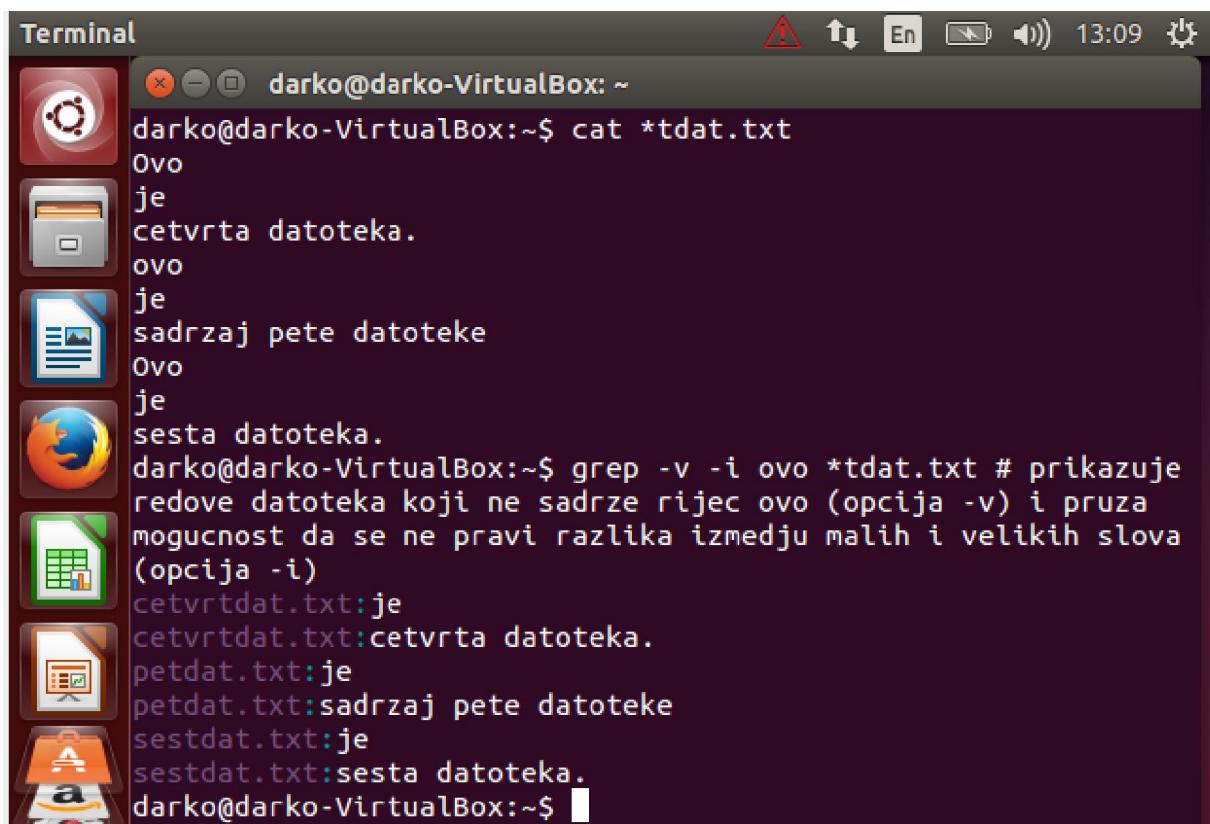
Slika 3.22.1. – Pretraživanje redova teksta uz pomoć grep



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark purple background and contains the following text:

```
darko@darko-VirtualBox: ~$ grep -c tty sablon.txt # ispisuje se broj redova koji sadrze tty
98
darko@darko-VirtualBox: ~$
```

Slika 3.22.2. – Primjer modifikovanja rezultata pretrage

A screenshot of an Ubuntu desktop environment. On the left, there is a dock with several icons: Dash, Home, File Explorer, Files, Firefox, LibreOffice Calc, LibreOffice Impress, and Amazon. The main window is a terminal window titled "Terminal". The terminal shows the following command-line session:

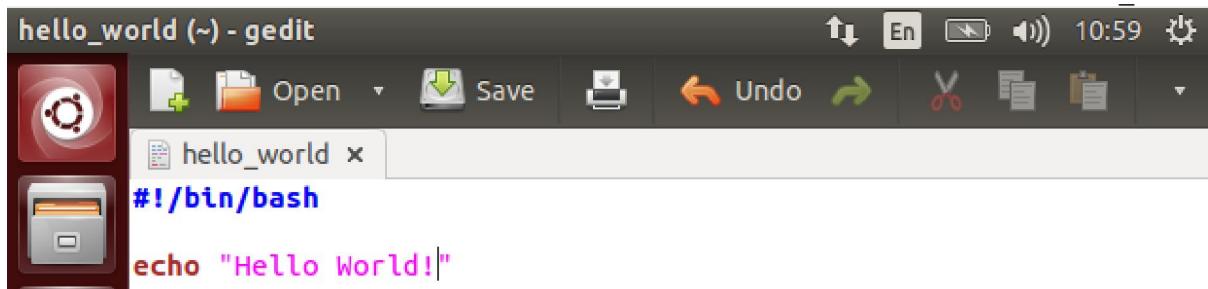
```
darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ cat *tdat.txt
Ovo
je
cetvrta datoteka.
ovo
je
sadrzaj pete datoteke
Ovo
je
sesta datoteka.
darko@darko-VirtualBox:~$ grep -v -i ovo *tdat.txt # prikazuje
redove datoteka koji ne sadrze rijec ovo (opcija -v) i pruza
mogucnost da se ne pravi razlika izmedju malih i velikih slova
(opcija -i)
cetvrt.dat.txt:je
cetvrt.dat.txt:cetvrta datoteka.
pet.dat.txt:je
pet.dat.txt:sadrzaj pete datoteke
sest.dat.txt:je
sest.dat.txt:sesta datoteka.
darko@darko-VirtualBox:~$
```

Slika 3.22.3. – Pretražuju se sve datoteke koje se završavaju na tdat.txt

4. PISANJE KOMANDNIH SKRIPTOVA

4.1. hello_world

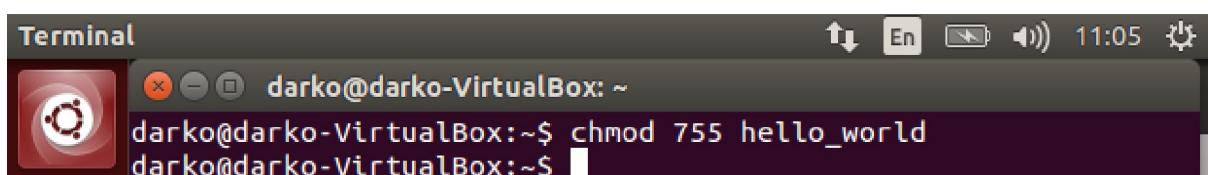
Komandni skript (engl. *shell script*) jeste niz komandi upisanih u datoteku; ljudska učitava te komande iz datoteke isto kao kada su otkucane u prozoru terminala [1]. Ne može se bilo koja tekstualna datoteka smatrati skriptom. Potrebno je podesiti ovlašćenja da bi skript mogao da se izvrši. Svaki komandni skript počinje sa konstrukcijom `#!` koja se naziva šebeng (engl. *shebang*). Šebeng se koristi da najavi ime programa koji se koristi za pokretanje komandi koje se nalaze u skriptu. U slučaju *bash* ljudske prvi red ima oblik: `#!/bin/bash`. Ovim se saopštava sistemu da se u direktorijumu bin nalazi program *bash* koji se koristi za pokretanje skripta. Pozdravnu poruku "Hello World!" treba otkucati u editoru teksta i sačuvati kao `hello_world`.



```
#!/bin/bash
echo "Hello World!"
```

Slika 4.1.1. – Ispisivanje pozdravne poruke na ekranu

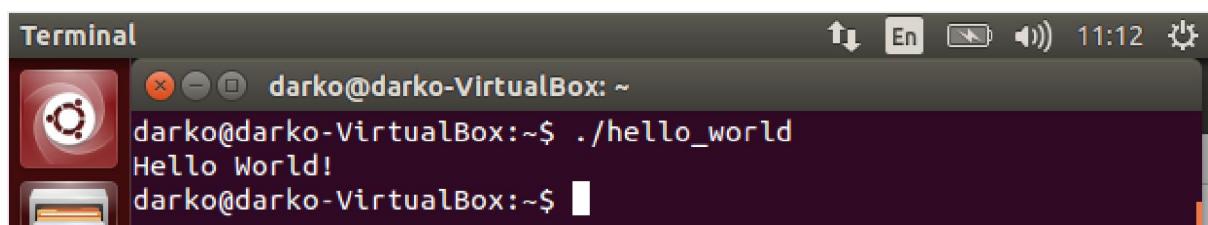
Sledeći korak je učiniti skript izvršnim. To se postiže unošenjem na komandnoj liniji `chmod 755 hello_world`. Komanda `chmod 755` omogućava da svi korisnici mogu da izvršavaju skript.



```
darko@darko-VirtualBox:~$ chmod 755 hello_world
```

Slika 4.1.2. – Unos koji omogućava da svi korisnici mogu da izvršavaju skript

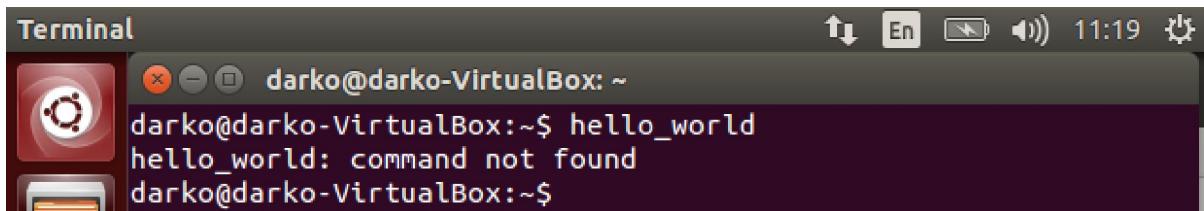
Pokretanje skripta vrši se unošenjem `./hello_world` na komandnoj liniji. Znak `.` je tekući radni direktorijum u kome se nalazi `/hello_world`.



```
darko@darko-VirtualBox:~$ ./hello_world
Hello World!
```

Slika 4.1.3. – Pokretanje skripta

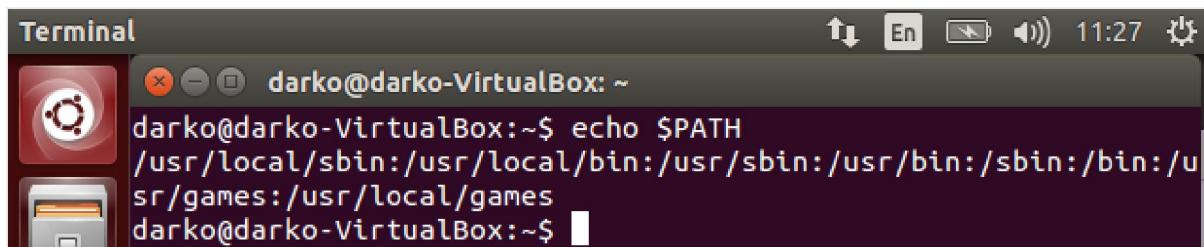
Ako se unese hello_world bez navođenja eksplisitne putanje dobija se sledeća poruka:



```
Terminal
darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ hello_world
hello_world: command not found
darko@darko-VirtualBox:~$
```

Slika 4.1.4. – Rezultat unosa skripta bez navođenja eksplisitne putanje

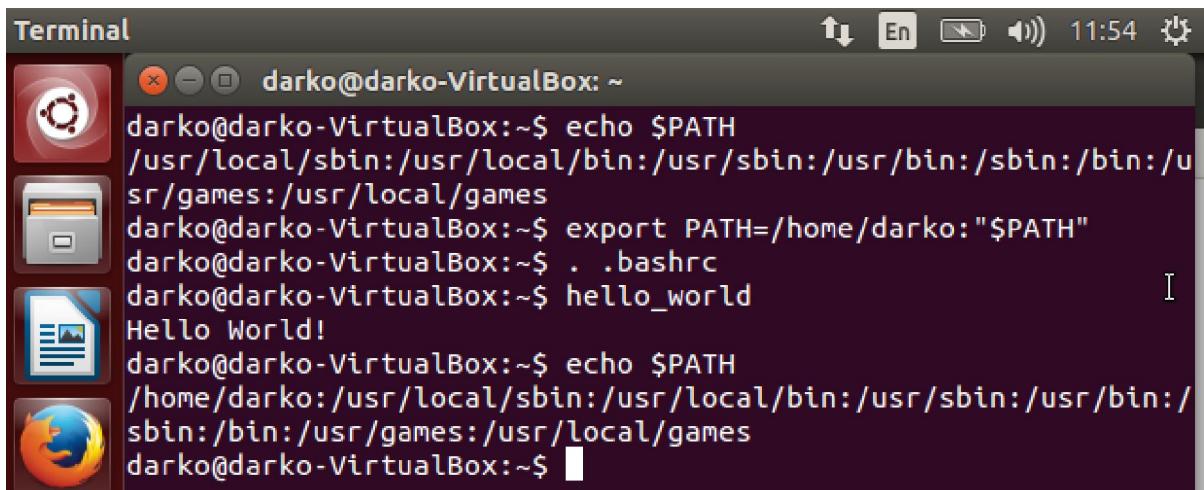
Ovo se dešava zato što ljudska unos hello_world doživljava kao i svaki drugi izvršni program. Skripta hello_world je sačuvana u matičnom direktorijumu. Spisak direktorijuma koje ljudska automatski pretražuje u potrazi za izvršnim programima nalazi se u promjenljivoj PATH. Oni su razdvojeni dvotačkama.



```
Terminal
darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
darko@darko-VirtualBox:~$
```

Slika 4.1.5. – Ispitivanje sadržaja promjenljive PATH

Da se ne bi navodila eksplisitna putanja moguće je matični direktorijum dodati u promjenljivu okruženja PATH. Unošenje se vrši komandom export na sledeći način: `export PATH=/home/darko:"$PATH"`. Ovo će važiti u sledećoj sesiji terminala. Da bismo omogućili da važi u trenutnoj potrebno je da ljudska ponovo pročita sistemsku datoteku .bashrc. Potrebno je unijeti `. .bashrc`. Znak `.` omogućava da se pročita datoteka .bashrc. Nakon toga nije potrebno pokretati terminal ispočetka jer je `/home/darko` uspješno dodat u promjenljivu PATH i skript hello_world će moći da se izvrši bez unošenja eksplisitne putanje.



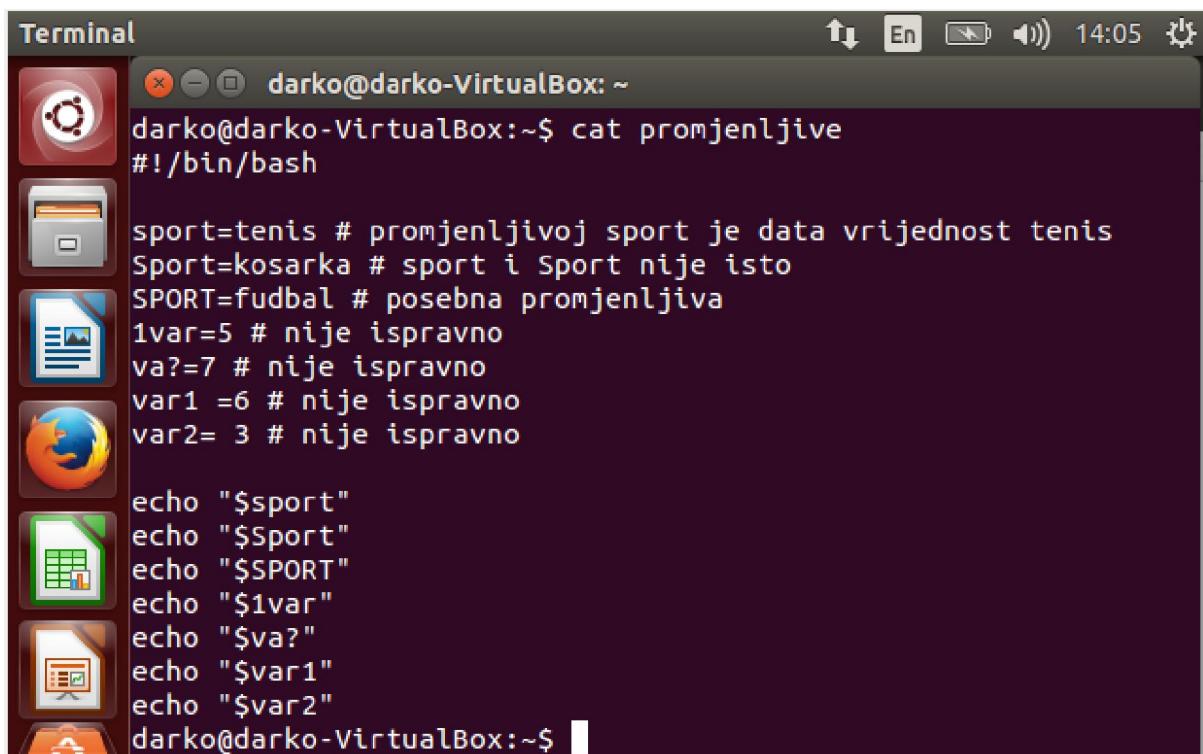
```
Terminal
darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
darko@darko-VirtualBox:~$ export PATH=/home/darko:"$PATH"
darko@darko-VirtualBox:~$ . .bashrc
darko@darko-VirtualBox:~$ hello_world
Hello World!
darko@darko-VirtualBox:~$ echo $PATH
/home/darko:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
darko@darko-VirtualBox:~$
```

Slika 4.1.6. – Dodavanje matičnog direktorijuma u promjenljivu okruženja PATH

4.2. Promjenljive

Promjenljiva predstavlja mali, imenovani dio memorije, kome može biti dodjeljena vrijednost. Ne postoji klasifikacija promjenljivih po tipovima jer sve promjenljive ljudska doživjava kao stringove. Kao što im samo ime ukazuje, njihova vrijednost se može mijenjati. Imena promjenljivih su "case sensitive". To znači da ime promjenljive *sport* nije isto što i ime *Sport*. Ako se na primjer nekoj promjenljivoj dodijeli vrijednost 5, ljudska tu promjenljiva doživjava kao string koji sadrži znak 5 a ne integer 5 kao što je to u nekim drugim programskim jezicima. Ime promjenljive mora biti jedinstveno i ne mogu dvije promjenljive da imaju isto ime. Postoje određena pravila za davanja imena promjenljivim. Ime promjenljive može sadržati sledeće znakove:

- a-z mala slova
- A-Z velika slova
- 0-9 cifre
- _ znak za podvlačenje



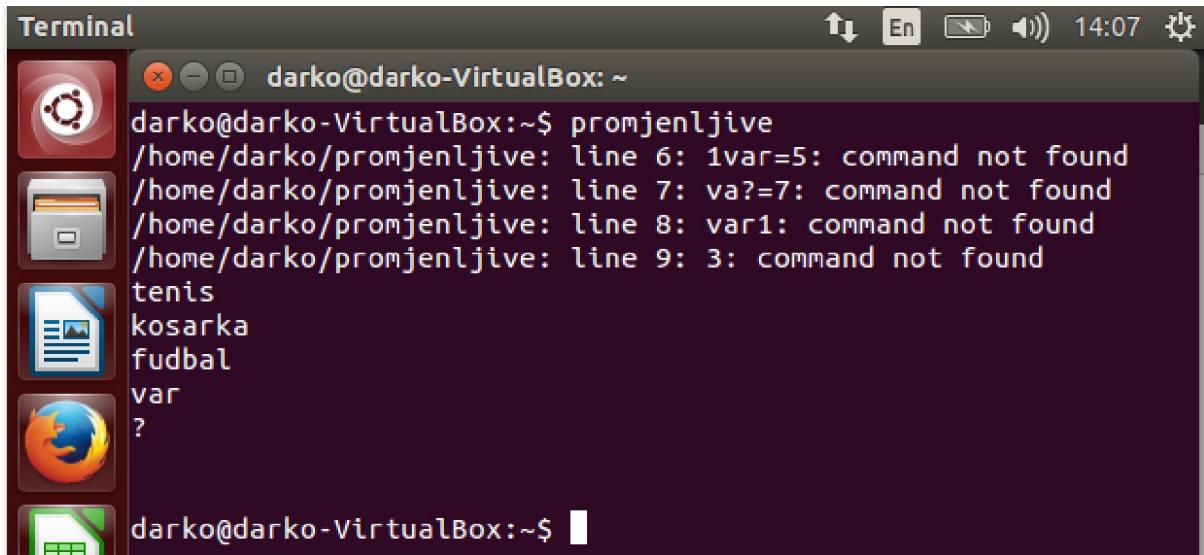
```
darko@darko-VirtualBox:~$ cat promjenljive
#!/bin/bash

sport=tenis # promjenljivoj sport je data vrijednost tenis
Sport=kosarka # sport i Sport nije isto
SPORT=fudbal # posebna promjenljiva
1var=5 # nije ispravno
va?=7 # nije ispravno
var1 =6 # nije ispravno
var2= 3 # nije ispravno

echo "$sport"
echo "$Sport"
echo "$SPORT"
echo "$1var"
echo "$va?"
echo "$var1"
echo "$var2"
darko@darko-VirtualBox:~$
```

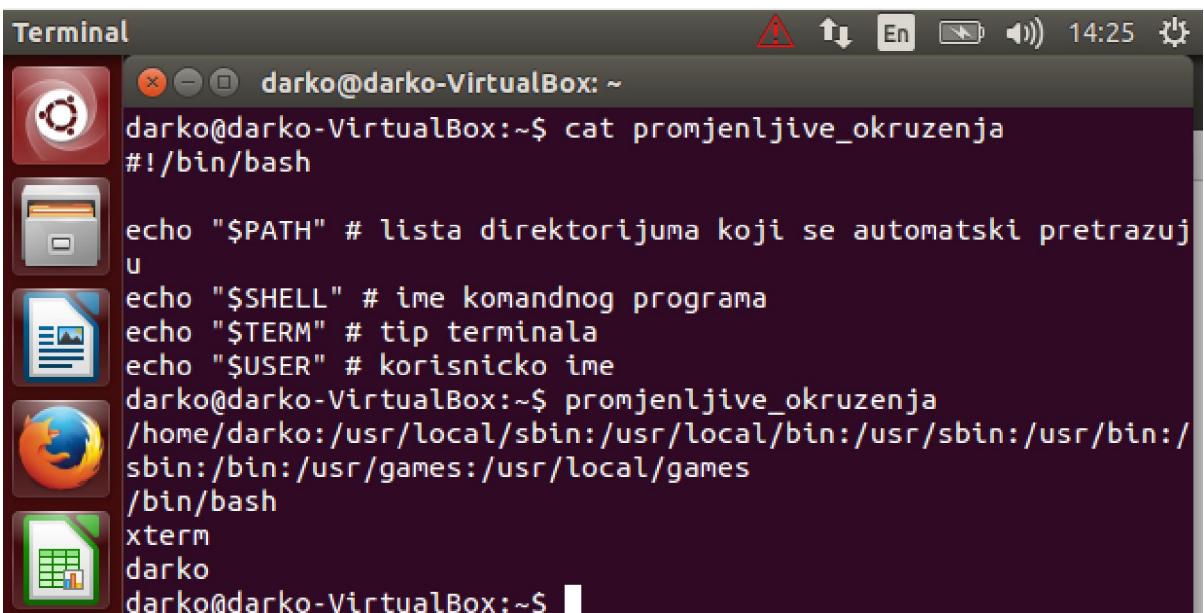
Slika 4.2.1. – Kod skripta koji ilustruje pravilnu i nepravilnu dodjelu vrijednosti promjenljivima

Ne smije biti razmaka sa bilo koje strane znaka = prilikom dodjele vrijednosti promjenljivoj. Ime promjenljive ne smije počinjati cifrom. Za dohvatanje vrijednosti koja je dodjeljena promjenljivoj koristi se znak \$. Ako se promjenljivoj dodijeli vrijednost koja sadrži razmake onda je potrebno tu vrijednost okružiti navodnicima. Promjenljive okruženja su specifičan tip promjenljivih i one su unaprijed postavljene.



```
darko@darko-VirtualBox:~$ promjenljive
/home/darko/promjenljive: line 6: 1var=5: command not found
/home/darko/promjenljive: line 7: va?=7: command not found
/home/darko/promjenljive: line 8: var1: command not found
/home/darko/promjenljive: line 9: 3: command not found
tenis
kosarka
fudbal
var
```

Slika 4.2.2. – Testiranje skripta za dodjelu vrijednosti promjenljivima



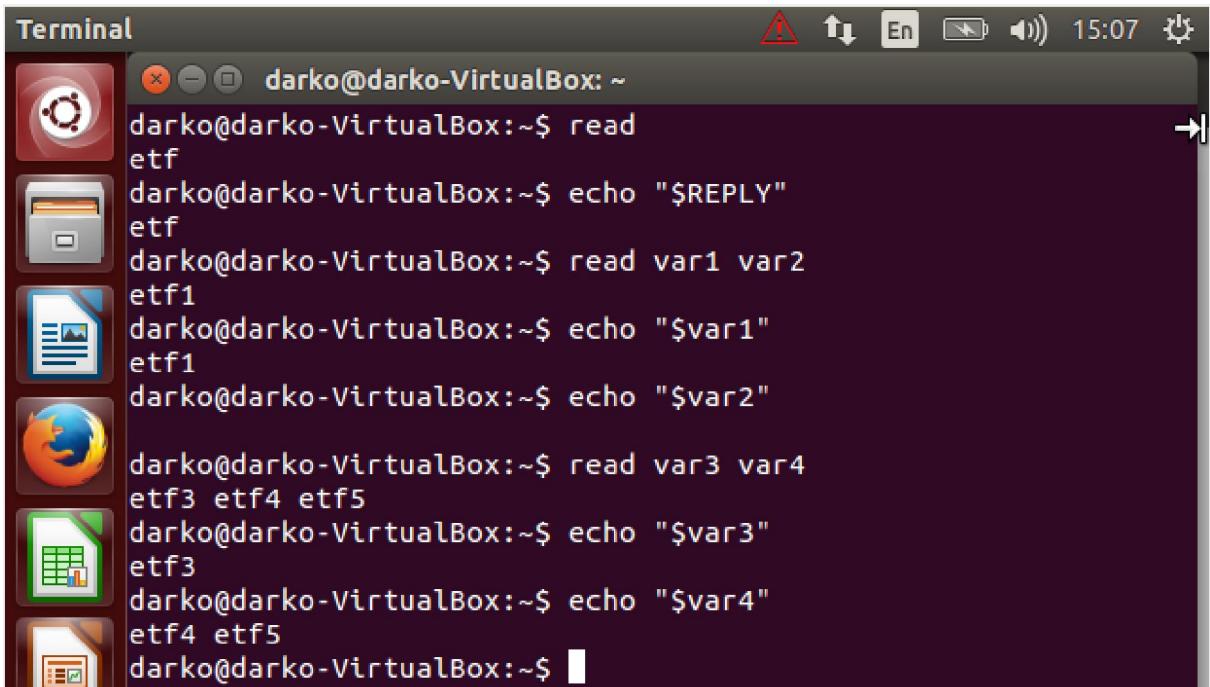
```
darko@darko-VirtualBox:~$ cat promjenljive_okruzenja
#!/bin/bash

echo "$PATH" # lista direktorijuma koji se automatski pretrazuju
echo "$SHELL" # ime komandnog programa
echo "$TERM" # tip terminala
echo "$USER" # korisnicko ime
darko@darko-VirtualBox:~$ promjenljive_okruzenja
/home/darko:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
/bin/bash
xterm
darko
darko@darko-VirtualBox:~$
```

Slika 4.2.3. – Ispitivanje sadržaja promjenljivih okruženja

4.3. read

Služi za čitanje standardnog ulaza (obično tastature) i smještanje informacija u promjenljive lјuske. U skriptovima se najčešće koristi da prima odgovore na pitanja koja postavlja skript. Ukoliko nije navedena promjenljiva u koju se smješta unos, taj unos se smješta u promjenljivu *REPLY*. Ako je broj unijetih parametara manji od broja promjenljivih, promjenljive koje su neiskorišćene neće sadržati ništa. Ako je broj unijetih parametara veći od broja promjenljivih parametra koji su "višak" smještaju se u poslednju promjenljivu.



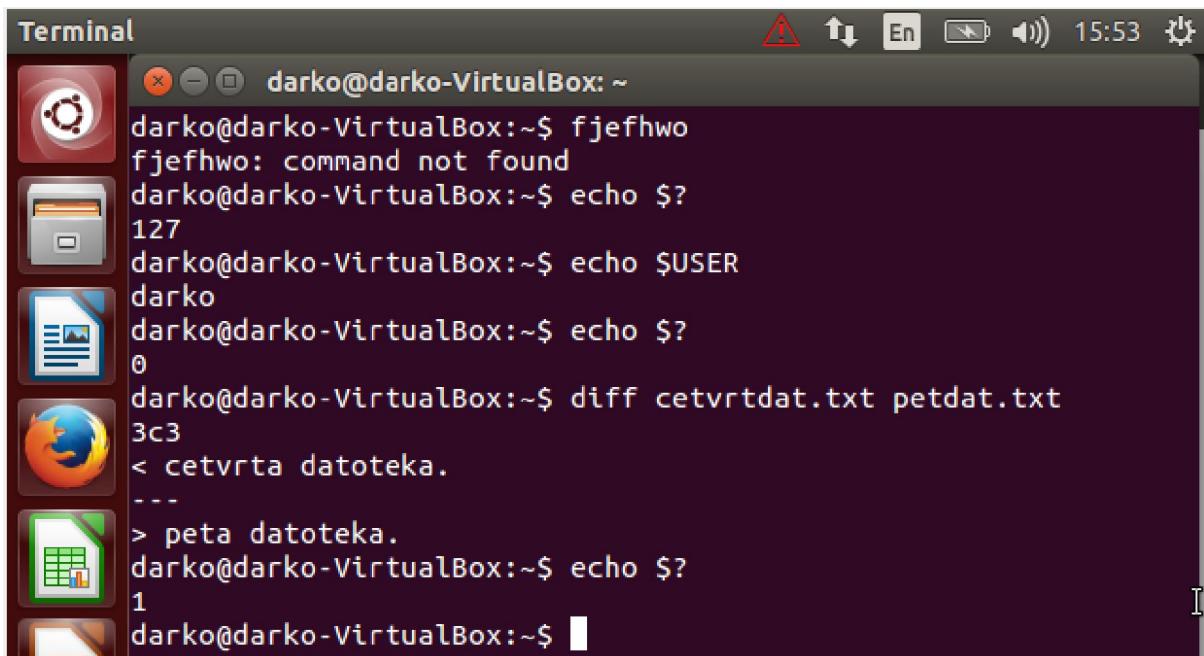
The screenshot shows a terminal window titled "Terminal" with a dark background. The window contains the following text:

```
darko@darko-VirtualBox:~$ read
etf
darko@darko-VirtualBox:~$ echo "$REPLY"
etf
darko@darko-VirtualBox:~$ read var1 var2
etf1
darko@darko-VirtualBox:~$ echo "$var1"
etf1
darko@darko-VirtualBox:~$ echo "$var2"
darko@darko-VirtualBox:~$ read var3 var4
etf3 etf4 etf5
darko@darko-VirtualBox:~$ echo "$var3"
etf3
darko@darko-VirtualBox:~$ echo "$var4"
etf4 etf5
darko@darko-VirtualBox:~$
```

Slika 4.3.1. – Upotreba naredbe read

4.4. Izlazni status

Nakon izvršenja bilo koje komande, nevidljivo se šalje poruka o njenoj uspješnosti u vidu izlaznog statusa. Izlazni status 0 upućuje na to da je komanda uspješno izvršena. Izlazni status različit od nule najčešće je znak da se desila neka greška. Međutim, to nije uvijek slučaj. Komanda diff daje izlazni status 0 ako su datoteke identične. Ako nisu identične, izlazni status je različit od nule iako nije došlo do greške. Izlazni status se ispituje unosom echo \$?.

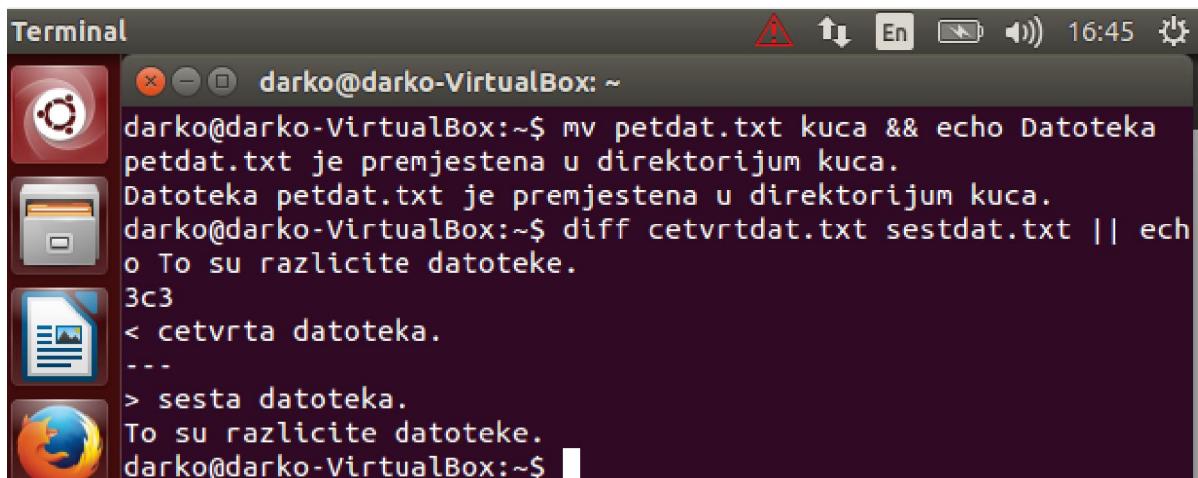


```
darko@darko-VirtualBox:~$ fjefhwo
fjefhwo: command not found
darko@darko-VirtualBox:~$ echo $?
127
darko@darko-VirtualBox:~$ echo $USER
darko
darko@darko-VirtualBox:~$ echo $?
0
darko@darko-VirtualBox:~$ diff cetvrtdat.txt petdat.txt
3c3
< cetvrta datoteka.
---
> peta datoteka.
darko@darko-VirtualBox:~$ echo $?
1
darko@darko-VirtualBox:~$
```

Slika 4.4.1. – Ispitivanje izlaznog statusa

4.5. Logički konstrukti && i ||

Moguće je postići da se neka naredba u skriptu izvrši samo ako je ispunjen određen uslov. Takvi uslovi su najčešće izlazni statusi drugih naredbi. Konstrukcija komanda1 && komanda2 omogućava da se komanda2 izvrši ako je izlazni status komande 1 jednak 0. Konstrukcija komanda1 || komanda2 omogućava da se komanda2 izvrši ako se komanda1 izvrši sa statusom koji je različit od nula.



```
darko@darko-VirtualBox:~$ mv petdat.txt kuca && echo Datoteka petdat.txt je premjestena u direktorijum kuca.
Datoteka petdat.txt je premjestena u direktorijum kuca.
darko@darko-VirtualBox:~$ diff cetvrtdat.txt sestdat.txt || echo To su razlicite datoteke.
3c3
< cetvrta datoteka.
---
> sesta datoteka.
To su razlicite datoteke.
darko@darko-VirtualBox:~$
```

Slika 4.5.1. – Primjer upotrebe logičkih konstrukata && i ||

4.6. if naredba

Ova naredba predstavlja mnogo moćniju konstrukciju u odnosu na gore navedene naredbe. Ima sledeći oblik:

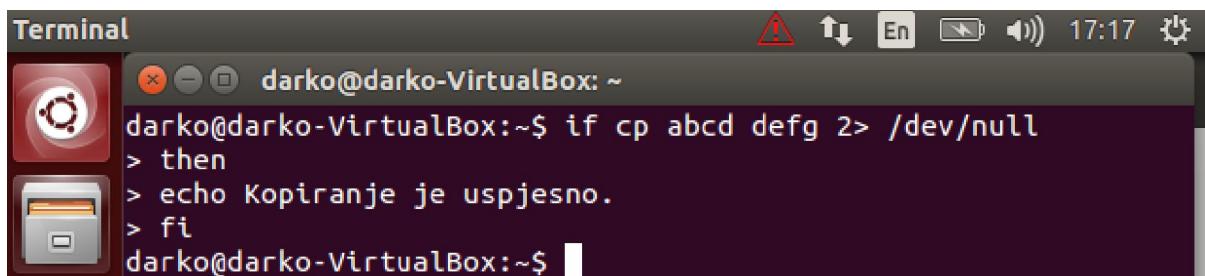
if komanda1

then

blok naredbi

fi# Ukoliko izlazni status nije nula prelazi se u ovaj red i terminira se if.

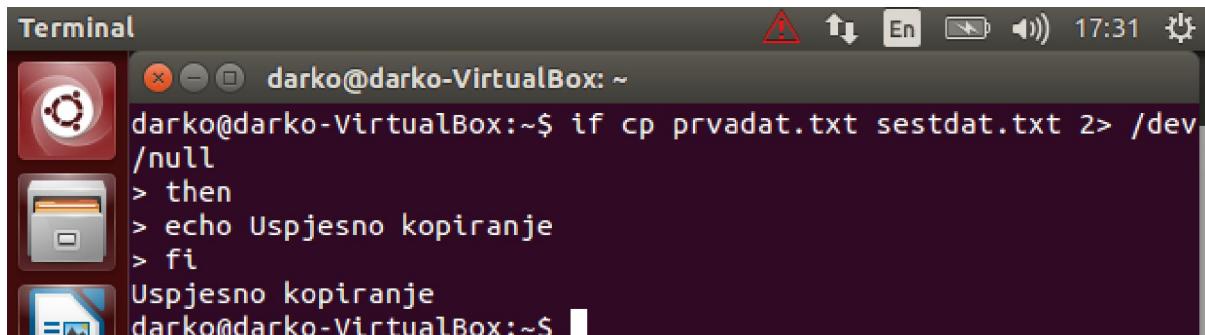
Na slici 4.6.1. ilustrovana je primjena if naredbe. Zadato je da se izvrši kopiranje datoteka, i da se poruka ako je kopiranje uspješno ispiše na ekranu. Kako kopiranje nije uspješno, jer su zadate nepostojeće datoteke, greška koja se desila preusmjerena je u /dev/null pa na ekranu nije prikazano ništa.



```
Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ if cp abcd defg 2> /dev/null
> then
> echo Kopiranje je uspjesno.
> fi
darko@darko-VirtualBox:~$
```

Slika 4.6.1. – Primjer neuspješnog kopiranja uz upotrebu if

Za kopiranje postojećih datoteka isti blok naredbi izgleda ovako:



```
Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ if cp prvadat.txt sestdat.txt 2> /dev/null
> then
> echo Uspjesno kopiranje
> fi
Uspjesno kopiranje
darko@darko-VirtualBox:~$
```

Slika 4.6.2. – Primjer uspješnog kopiranja uz upotrebu if

4.7. else

Koristi se sa if naredbom i omogućava da se ukoliko komanda u uslovu ne vrati null izlaz zada novi blok naredbi. Ima oblik:

```

if komanda1 # ukoliko je izlazni status komanda1 0 prelazi se u prvi blok naredbi

then

prvi blok naredbi

else

drugi blok naredbi # ukoliko je izlaz komanda1 nenulti prelazi se u drugi blok naredbi

```

fi

```

Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ if cp prvadat.txt sestdat.txt
> then
> echo Uspjesno!
> else
> echo Neuspjesno!
> fi
Uspjesno!
darko@darko-VirtualBox:~$ 

```

Slika 4.7.1. – Upotreba else naredbe

4.8. elif

Često je potrebno zadati više od dva bloka naredbi. To omogućava elif. Ima sledeći oblik:

if komanda1

then

prvi blok naredbi

elif komanda2

then

drugi blok naredbi

else

treci blok naredbi

f1

Ukoliko *komanda1* u if-u daje nenulti izlazni status prelazi se na elif. Ako *komanda2* u elif-u daje nenulti izlazni status prelazi se na else. Naredba elif se može ponavljati beskonačno puta, dok je naredbu else moguće iskoristiti samo jednom.

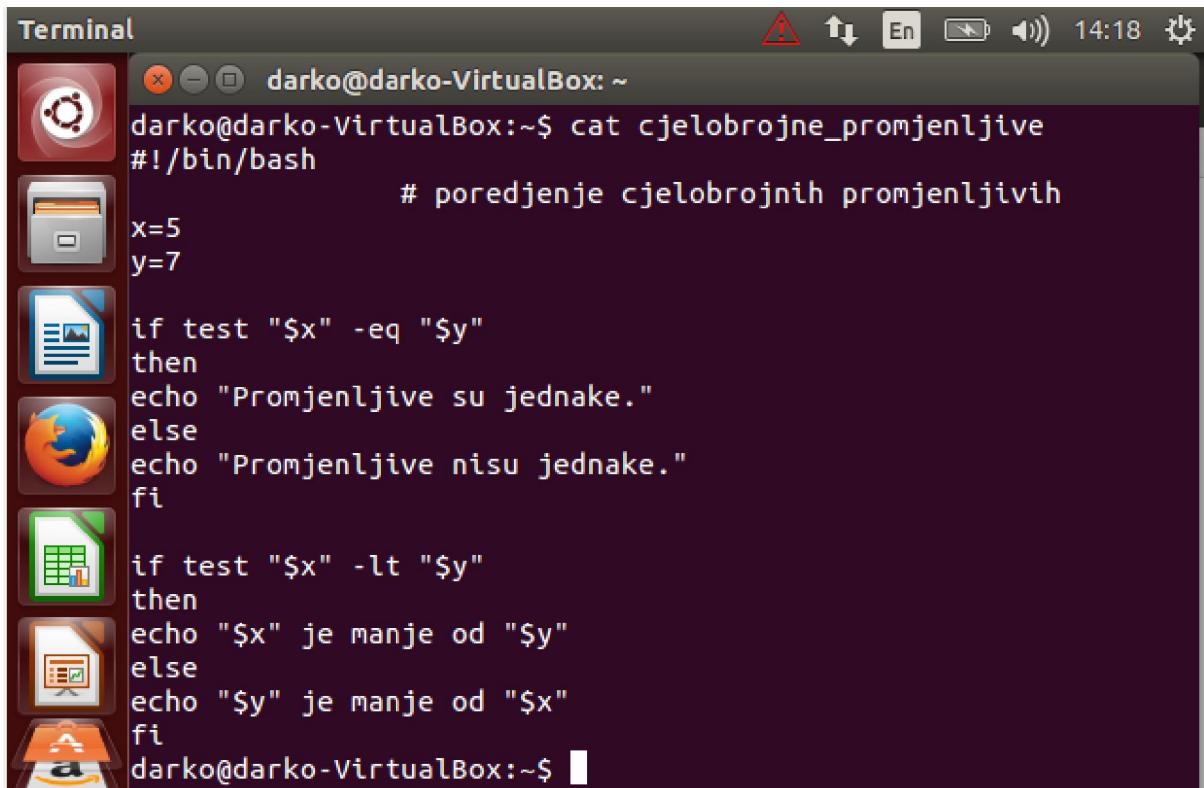
4.9. test

Koristii se sa if naredbom. Ne daje nikakav rezultat na ekranu već if koristi izlaz ove naredbe kao svoj uslov. Naredba test najčešće poredi dvije vrijednosti i generiše nulu za uspješno poređenje i nenulti izlaz za neuspješno poređenje. Iako ljska ne poznaje tipove podataka već sve tretira kao niz znakova moguće je izvršiti određene operacije poređenja nad promjenljivima. Od ključne važnosti je da li se promjenljiva sastoji samo od cifara ili ne.

4.9.1. *Ispitivanje cjelobrojnih promjenljivih*

Operacije poređenja koje su dozvoljene:

- -eq jednako
- -ne nije jednako
- -le manje ili jednako
- -lt manje
- -ge veće ili jednako
- -gt veće

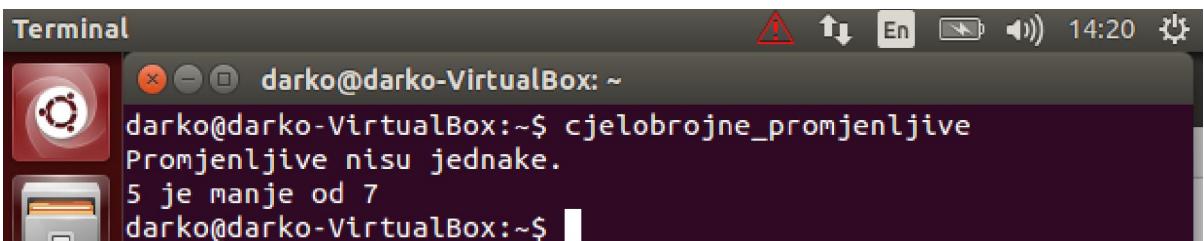


```
darko@darko-VirtualBox:~$ cat cjelobrojne_promjenljive
#!/bin/bash
# poređenje cjelobrojnih promjenljivih
x=5
y=7

if test "$x" -eq "$y"
then
echo "Promjenljive su jednake."
else
echo "Promjenljive nisu jednake."
fi

if test "$x" -lt "$y"
then
echo "$x" je manje od "$y"
else
echo "$y" je manje od "$x"
fi
darko@darko-VirtualBox:~$
```

Slika 4.9.1.1. – Ilustracija poređenja cjelobrojnih promjenljivih



```
darko@darko-VirtualBox:~$ cjelobrojne_promjenljive
Promjenljive nisu jednake.
5 je manje od 7
darko@darko-VirtualBox:~$
```

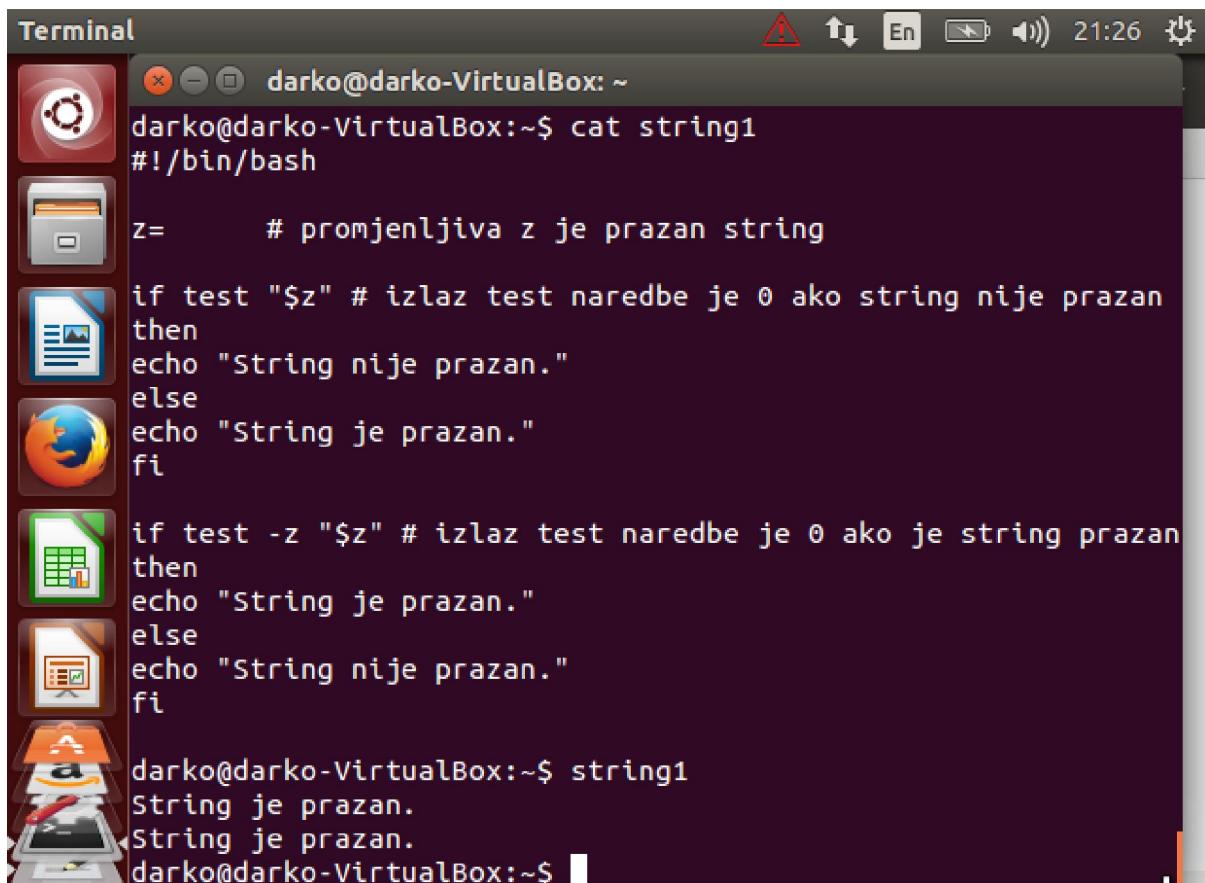
Slika 4.9.1.2. – Rezultat poređenja cjelobrojnih promjenljivih

4.9.2. Rad sa stringovima

Stringovi predstavljaju nizove bilo kojih znakova i ljudska dozvoljava određene operacije njihovog poređenja. Neke od njih su:

- `-n string` ispituje da li je dužina stringa veća od nule
- `-z string` ispituje da li je dužina stringa jednaka nuli
- `=` ispituje da li su stringovi jednaki

- `!=` ispituje da li su stringovi različiti
- `> , <` ispituje koji od dva stringa prednjači u ASCII tabeli



```

Terminal
darko@darko-VirtualBox:~$ cat string1
#!/bin/bash

z=      # promjenljiva z je prazan string

if test "$z" # izlaz test naredbe je 0 ako string nije prazan
then
echo "String nije prazan."
else
echo "String je prazan."
fi

if test -z "$z" # izlaz test naredbe je 0 ako je string prazan
then
echo "String je prazan."
else
echo "String nije prazan."
fi

darko@darko-VirtualBox:~$ string1
String je prazan.
String je prazan.
darko@darko-VirtualBox:~$ 
```

Slika 4.9.2.1. – Kod i rezultat skripta koji ilustruje operacije nad praznim stringom

```
darko@darko-VirtualBox: ~
#!/bin/bash

a=]
b=[

if test "$a" = "$b" # sa obje strane = mora biti razmak
then
echo "Stringovi su jednaki."
else
echo "Stringovi nisu jednaki."
fi

if test "$a" \> "$b" # vrsti poredjenje ASCII vrijednosti
then
echo Znak "$a" prednjaci u odnosu na "$b".
else
echo Znak "$b" prednjaci u odnosu na "$a".
fi

darko@darko-VirtualBox:~$ string2
Stringovi nisu jednaki.
Znak ] prednjaci u odnosu na [ .
darko@darko-VirtualBox:~$
```

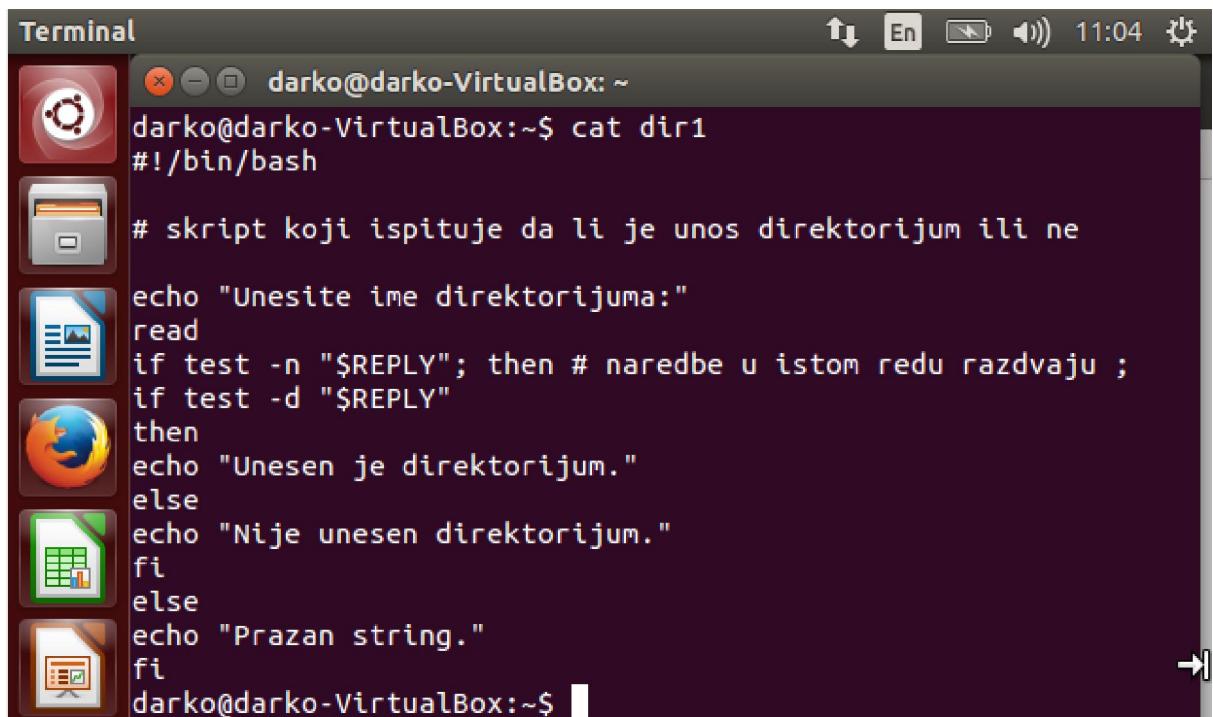
Slika 4.9.2.2. – Kod i rezultat skripta koji ilustruje operacije nad stringovima | i [. Ispituje se da li su jednaki i koji od stringova | ili [prednjači u ASCII tabeli. Uz < i > se koristiti \ jer uklanja redirekciju kod ovih znakova i omogućava da se oni koriste za poređenje.

4.9.3. Ispitivanje datoteka

Neke od naredbi koje se koriste za ispitivanje datoteka su:

- -e datoteka ispituje da li datoteka postoji
 - -f datoteka ispituje da li datoteka postoji i predstavlja običnu datoteku
 - -s datoteka ispituje da li datoteka postoji i veća je od nule
 - -x datoteka ispituje da li datoteka postoji i može se izvršavati/pretraživati
 - -r datoteka ispituje da li datoteka postoji i može se čitati

- -d datoteka ispituje da li datoteka postoji i predstavlja direktorijum

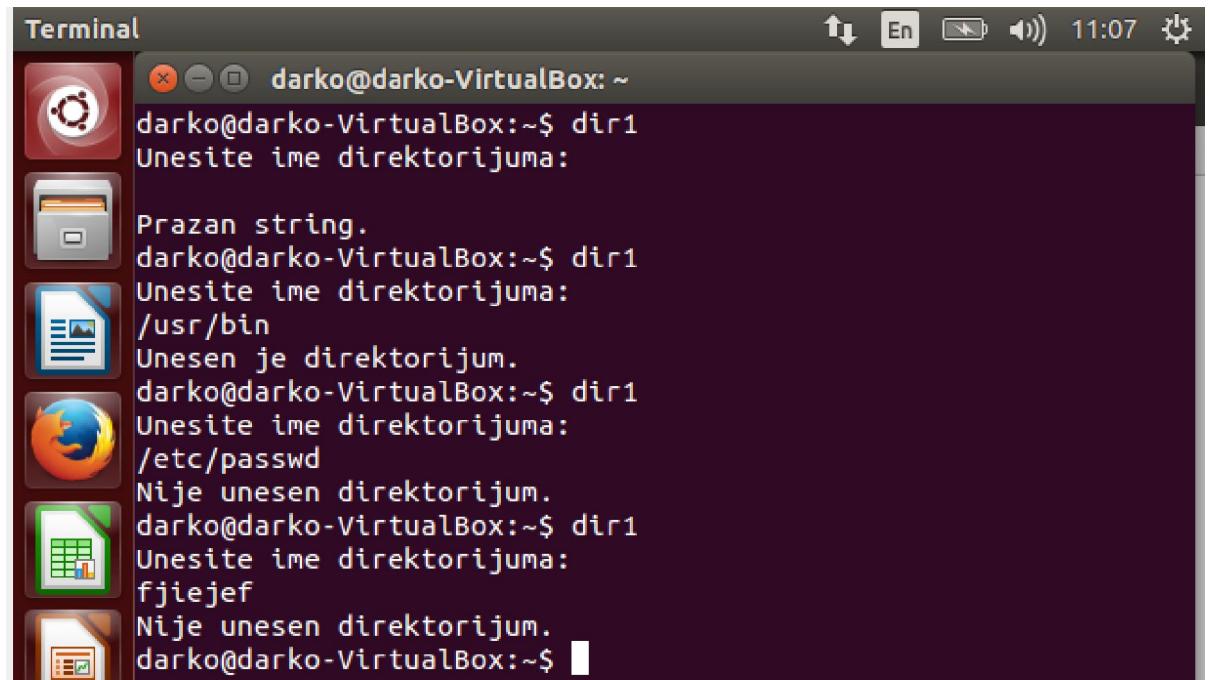


```
darko@darko-VirtualBox:~$ cat dir1
#!/bin/bash

# skript koji ispituje da li je unos direktorijum ili ne

echo "Unesite ime direktorijuma:"
read
if test -n "$REPLY"; then # naredbe u istom redu razdvaju ;
if test -d "$REPLY"
then
echo "Unesen je direktorijum."
else
echo "Nije unesen direktorijum."
fi
else
echo "Prazan string."
fi
darko@darko-VirtualBox:~$
```

Slika 4.9.3.1. – Skript koji ispituje da li je dat unos direktorijum ili ne



```
darko@darko-VirtualBox:~$ dir1
Unesite ime direktorijuma:

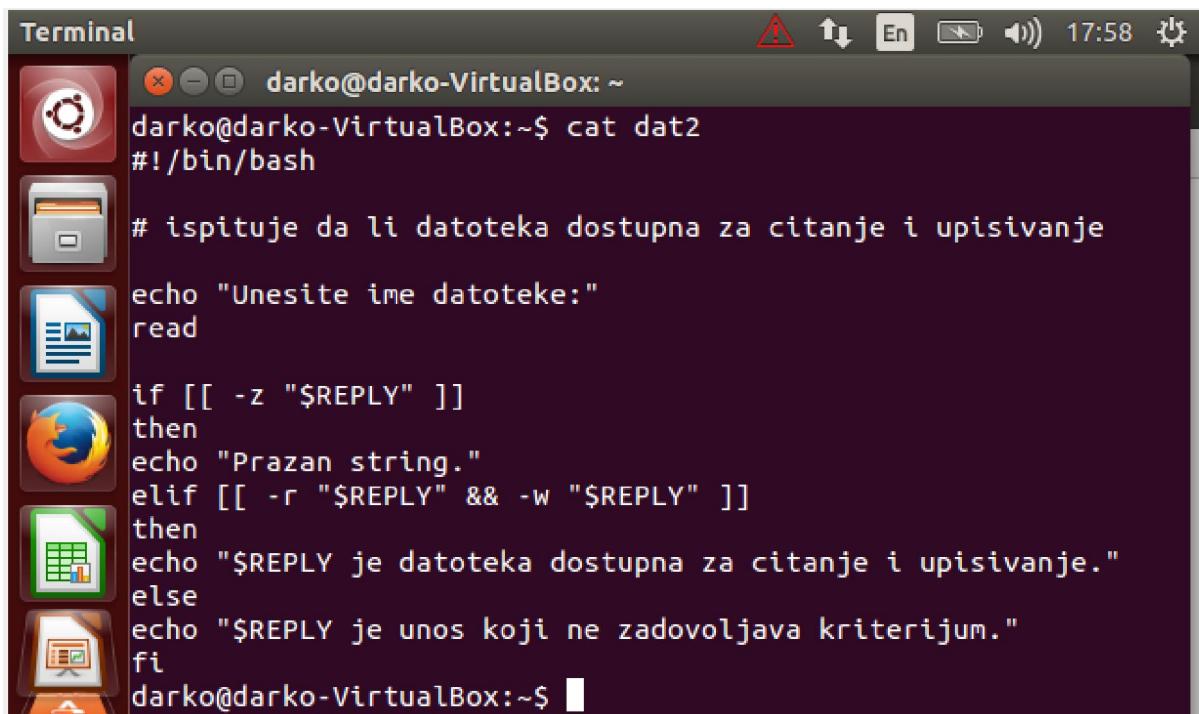
Prazan string.
darko@darko-VirtualBox:~$ dir1
Unesite ime direktorijuma:
/usr/bin
Unesen je direktorijum.
darko@darko-VirtualBox:~$ dir1
Unesite ime direktorijuma:
/etc/passwd
Nije unesen direktorijum.
darko@darko-VirtualBox:~$ dir1
Unesite ime direktorijuma:
fjiejef
Nije unesen direktorijum.
darko@darko-VirtualBox:~$
```

Slika 4.9.3.2. – Testiranje skripta koji provjerava da li je unos direktorijum ili ne

Ekvivalent zapisu test je naredba []. Umjesto test *izraz* može se koristiti [*izraz*]. Oba zapisa daju identični rezultat a razlika je jedino u sintaksi.

4.10. Konstrukcija [[]]

Konstrukcija [[*izraz*]] predstavlja poboljšanu verziju naredbe test. Dozvoljena je na primjer upotreba operatora && i || u sklopu [[*izraz*]] [3]. Navedeni operatori u sklopu naredbe test uzrokuju sintaksnu grešku [3]. Bitno poboljšanje predstavlja i mogućnost upotrebe proširenih regularnih izraza. Moguće je porediti zadati znakovni niz sa proširennim regularnim izrazom uz pomoć =~. U zavisnosti od uspješnosti poređenja vraća se 0 ili neka druga vrijednost.



```
darko@darko-VirtualBox:~$ cat dat2
#!/bin/bash

# ispituje da li datoteka dostupna za citanje i upisivanje

echo "Unesite ime datoteke:"
read

if [[ -z "$REPLY" ]]
then
echo "Prazan string."
elif [[ -r "$REPLY" && -w "$REPLY" ]]
then
echo "$REPLY je datoteka dostupna za citanje i upisivanje."
else
echo "$REPLY je unos koji ne zadovoljava kriterijum."
fi
darko@darko-VirtualBox:~$
```

Slika 4.10.1. – Kod skripta koji provjerava da li je neka datoteka istovremeno dostupna korisniku za čitanje i upisivanje



```
Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ dat2
Unesite ime datoteke:
fkoe
fkoe je unos koji ne zadovoljava kriterijum.
darko@darko-VirtualBox:~$ dat2
Unesite ime datoteke:

Prazan string.
darko@darko-VirtualBox:~$ dat2
Unesite ime datoteke:
/home/darko
/home/darko je datoteka dostupna za citanje i upisivanje.
darko@darko-VirtualBox:~$ dat2
Unesite ime datoteke:
string1
string1 je datoteka dostupna za citanje i upisivanje.
darko@darko-VirtualBox:~$
```

Slika 4.10.2. – Testiranje skripta koji provjera da li je neka datoteka dostupna korisniku za čitanje i upisivanje.
Važno je napomenuti da se i direktorijumi u unixolikim sistemima smatraju za datoteke koje sadrže druge
datoteke i direktorijume.



```
Terminal darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ cat int1
#!/bin/bash

k=-9; m=-1; n=1; r=9 # naredbe u istom redu se razdvajaju sa ;

echo "Unesite cijelobrojnu vrijednost u prvom intervalu od -9 do
-1 ili u drugom intervalu od 1 do 9:"
read x
if [[ "$x" =~ ^-?[1-9]$ ]]; then
if [[ "$x" -ge "$k" && "$x" -le "$m" ]]
then echo "Broj $x je u prvom intervalu."
elif [[ "$x" -ge "$n" && "$x" -le "$r" ]]
then echo "Broj $x je u drugom intervalu."
fi
else echo "Unos je pogresan."
fi
darko@darko-VirtualBox:~$
```

Slika 4.10.3. – Kod skripta koji traži da se unese cijeli broj u ponuđenim intervalima. Važno je napomenuti da je
^-?[1-9]\$ šablon koji predstavlja brojeve koji se nalaze u ponuđenim intervalima. Znak ^ predstavlja početak
šablonu, \$ je kraj. Dio -? opcionalo pojavljivanje znaka -, a [1-9] predstavlja brojeve od 1 do 9.

```
darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ int1
Unesite cijelobrojnu vrijednost u prvom intervalu od -9 do -1 ili u drugom intervalu od 1 do 9:
-8
Broj -8 je u prvom intervalu.
darko@darko-VirtualBox:~$ int1
Unesite cijelobrojnu vrijednost u prvom intervalu od -9 do -1 ili u drugom intervalu od 1 do 9:
0
Unos je pogresan.
darko@darko-VirtualBox:~$ int1
Unesite cijelobrojnu vrijednost u prvom intervalu od -9 do -1 ili u drugom intervalu od 1 do 9:
5
Broj 5 je u drugom intervalu.
darko@darko-VirtualBox:~$
```

Slika 4.10.4. – Testiranje skripta, ukoliko unos zadovoljava šablon brojeva u ponudenim intervalima vrši se utvrđivanje kom intervalu pripada. Ako ne, ispisuje se poruka o pogrešnom unosu.

4.11. case naredba

Konstrukcija koja služi, da u slučaju kada postoji mnogo elif naredbi, pojednostavi pisanje.
Ova naredba ima sledeći oblik:

```
case "$var" in
```

```
    prvi_slucaj)
```

```
        blok1
```

```
;;
```

```
    drugi_slucaj)
```

```
        blok2
```

```
;;
```

```
    treći_slucaj)
```

```
        blok3
```

```
;;
```

```
    četvrti_slucaj)
```

```
        blok4
```

```

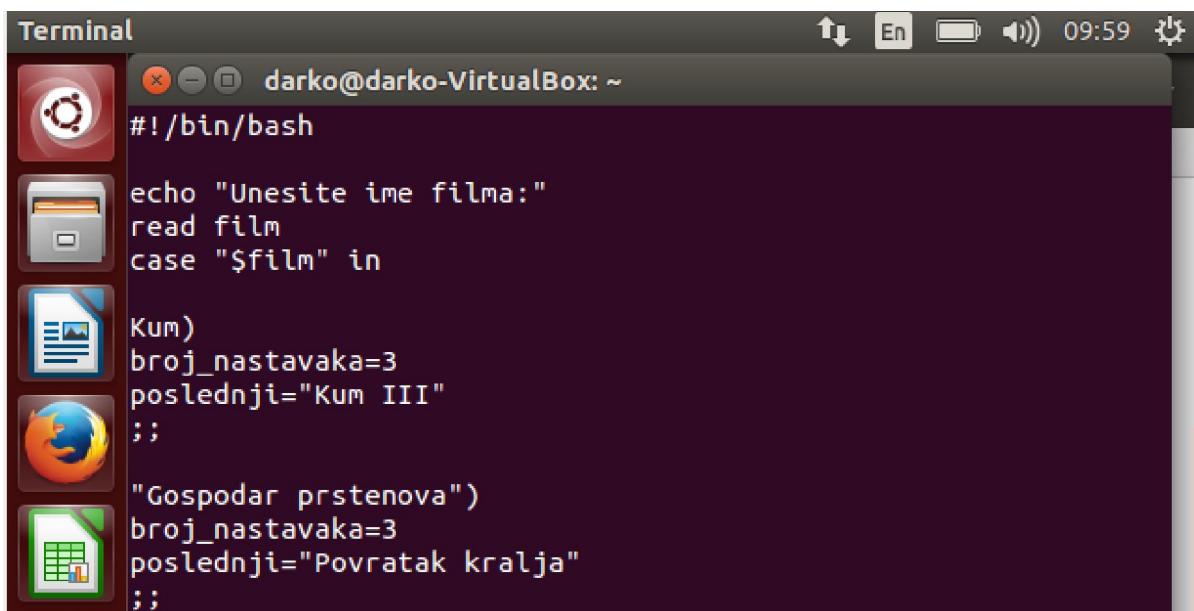
;;
.

.

nti_slucaj)
blokn

;;
esac
```

Sadržaj promjenljive *var* se poredi sa slučajevima od 1 do n i na osnovu uspješnosti se izvršava odgovarajući blok naredbi.

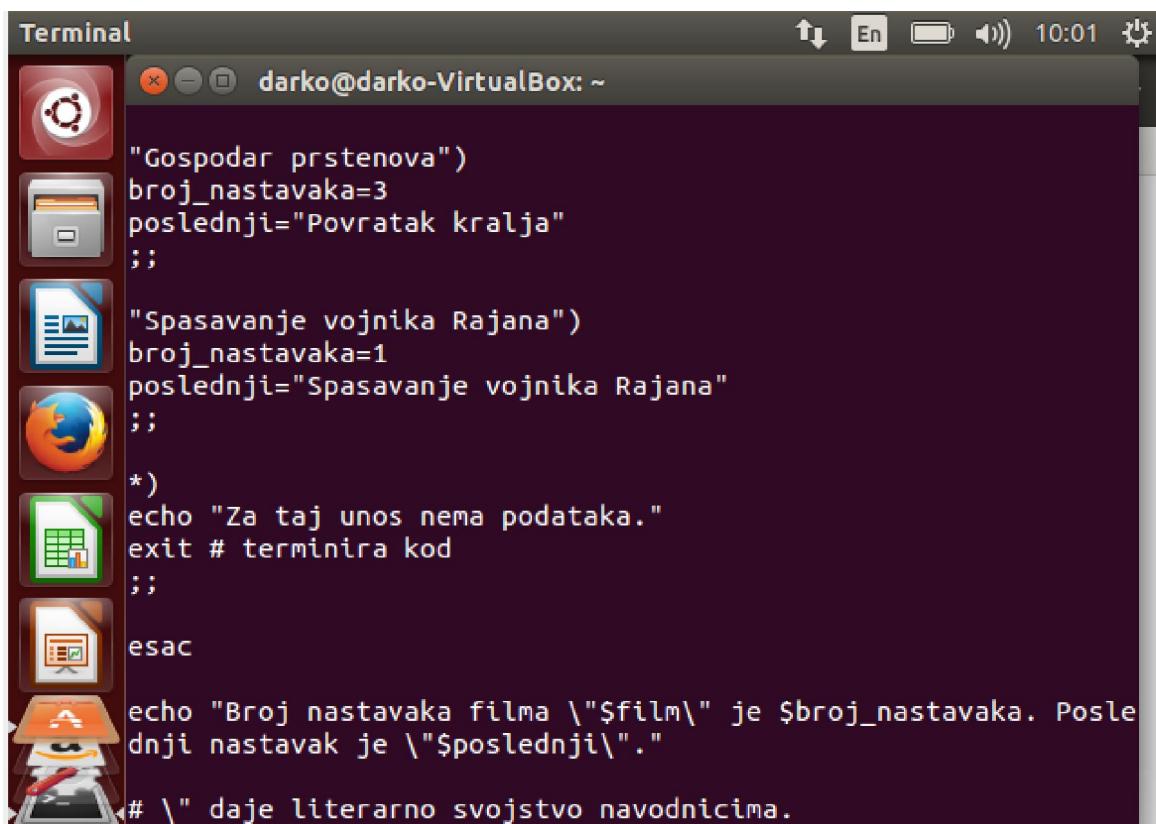


```

Terminal
darko@darko-VirtualBox: ~
#!/bin/bash

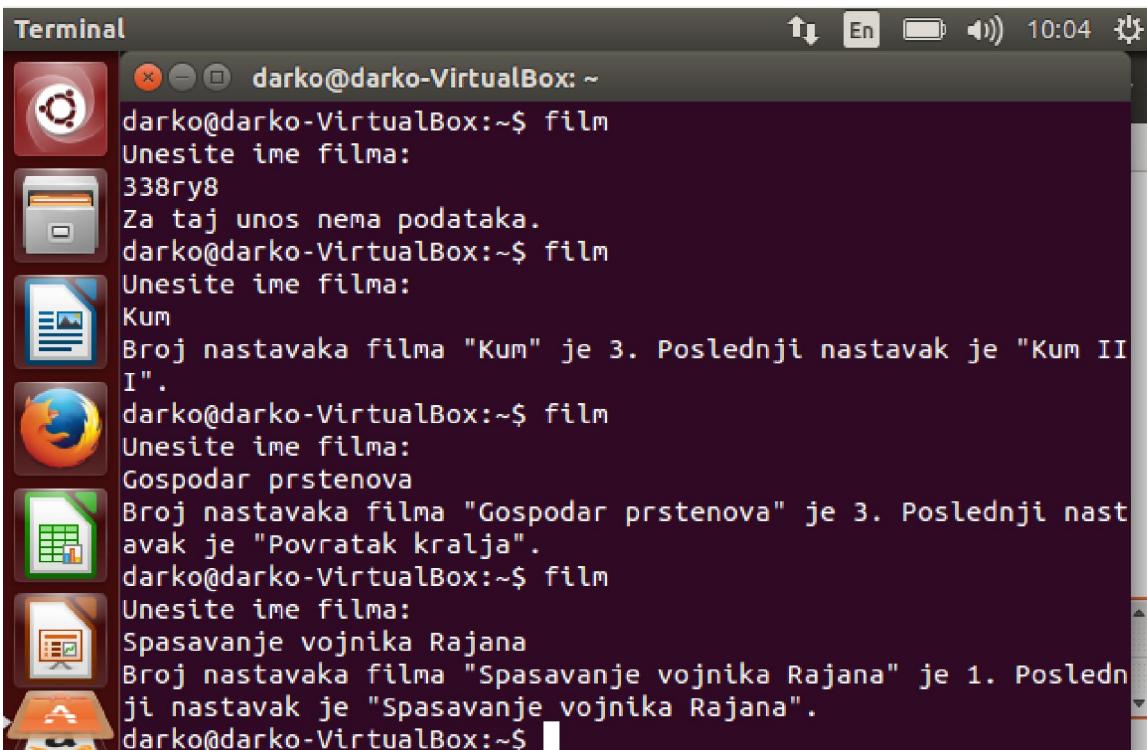
echo "Unesite ime filma:"
read film
case "$film" in
    Kum)
        broj_nastavaka=3
        poslednji="Kum III"
        ;;
    "Gospodar prstenova")
        broj_nastavaka=3
        poslednji="Povratak kralja"
        ;;
    *) echo "Ne poznati film"
        broj_nastavaka=1
        poslednji=$film
        ;;
esac
```

Slika 4.11.1. – Prvi dio skripta film koji od korisnika traži da unese ime filma. Taj unos se smješta u promjenljivu *film*. Sadržaj promjenljive *film* se poredi sa filmovima u bazi i na osnovu rezultata poređenja na ekranu se daje adekvatan ispis.



```
darko@darko-VirtualBox: ~
"Gospodar prstenova")
broj_nastavaka=3
poslednji="Povratak kralja"
;;
"Spasavanje vojnika Rajana")
broj_nastavaka=1
poslednji="Spasavanje vojnika Rajana"
;;
*)
echo "Za taj unos nema podataka."
exit # terminira kod
;;
esac
echo "Broj nastavaka filma \"$film\" je $broj_nastavaka. Posle
dnji nastavak je \"$poslednji\"."
# \" daje literarno svojstvo navodnicima.
```

Slika 4.11.2. – Nastavak koda skripta film



```
darko@darko-VirtualBox: ~$ film
Unesite ime filma:
338ry8
Za taj unos nema podataka.
darko@darko-VirtualBox:~$ film
Unesite ime filma:
Kum
Broj nastavaka filma "Kum" je 3. Poslednji nastavak je "Kum II
I".
darko@darko-VirtualBox:~$ film
Unesite ime filma:
Gospodar prstenova
Broj nastavaka filma "Gospodar prstenova" je 3. Poslednji nast
avak je "Povratak kralja".
darko@darko-VirtualBox:~$ film
Unesite ime filma:
Spasavanje vojnika Rajana
Broj nastavaka filma "Spasavanje vojnika Rajana" je 1. Posledn
ji nastavak je "Spasavanje vojnika Rajana".
darko@darko-VirtualBox:~$
```

Slika 4.11.3. - Testiranje skripta film

4.12.while

Ova petlja izvršava blok naredbi sve dok je njen uslov zadovoljen. Ima sledeći oblik:

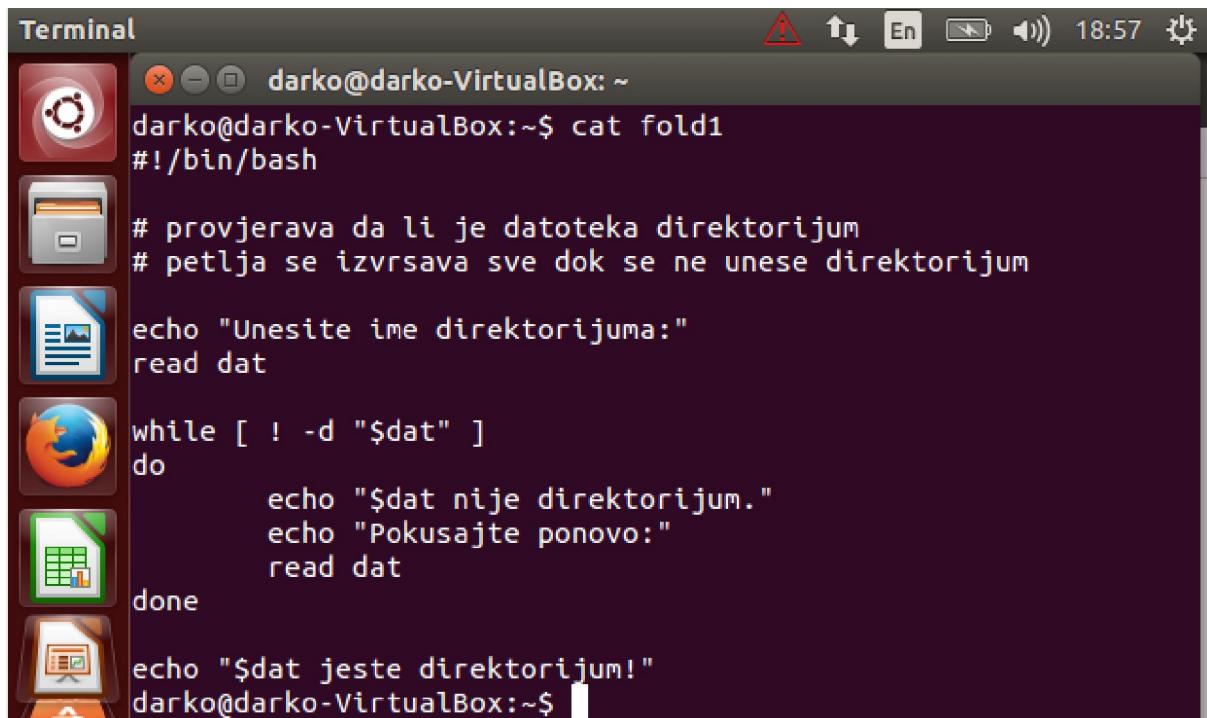
```
while komanda      # sve dok je izlazni status komanda nula ponavljaće se petlja
do
    blok naredbi
done
```

Kao i if naredba najčešće se koristi sa test.

```
while test izraz    # sve dok je izlazni status test izraz nula izvršava se
do
    blok naredbi
done
```

Može da se koristi i sa svestranom [[]] konstrukcijom [3]. Tada ima oblik:

```
while [[ izraz ]]
do
    blok naredbi
done
```



```
darko@darko-VirtualBox:~$ cat fold1
#!/bin/bash

# provjerava da li je datoteka direktorijum
# petlja se izvrsava sve dok se ne unese direktorijum

echo "Unesite ime direktorijuma:"
read dat

while [ ! -d "$dat" ]
do
    echo "$dat nije direktorijum."
    echo "Pokusajte ponovo:"
    read dat
done

echo "$dat jeste direktorijum!"
```

Slika 4.12.1. – Kod skripta fold1 koji od korisnika traži da unese direktorijum. Petlja se izvršava sve dok korisnik ne unese direktorijum.

```
darko@darko-VirtualBox: ~$ fold1
Unesite ime direktorijuma:
string1
string1 nije direktorijum.
Pokusajte ponovo:
film
film nije direktorijum.
Pokusajte ponovo:
/dev/null
/dev/null nije direktorijum.
Pokusajte ponovo:
/usr
/usr jeste direktorijum!
darko@darko-VirtualBox:~$
```

Slika 4.12.2. – Testiranje skripta fold1 koji od korisnika traži da unese direktorijum

4.13. for

Ova petlja ima sledeći oblik:

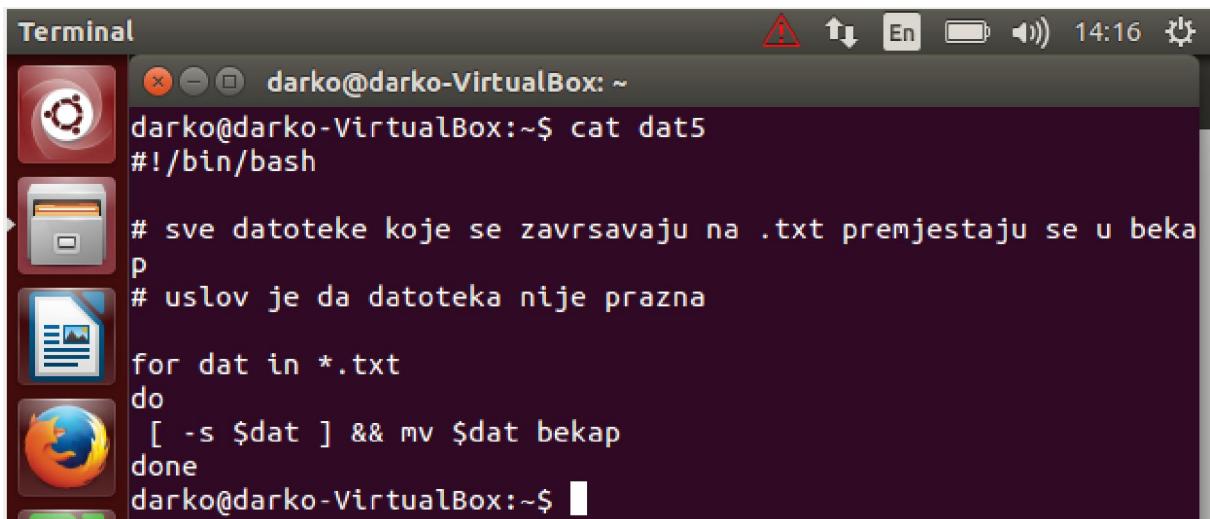
for var in list # promjenljiva var preuzima sve vrijednosti iz list

do

blok naredbi

done

Najčešće se koristi u radu sa datotekama.



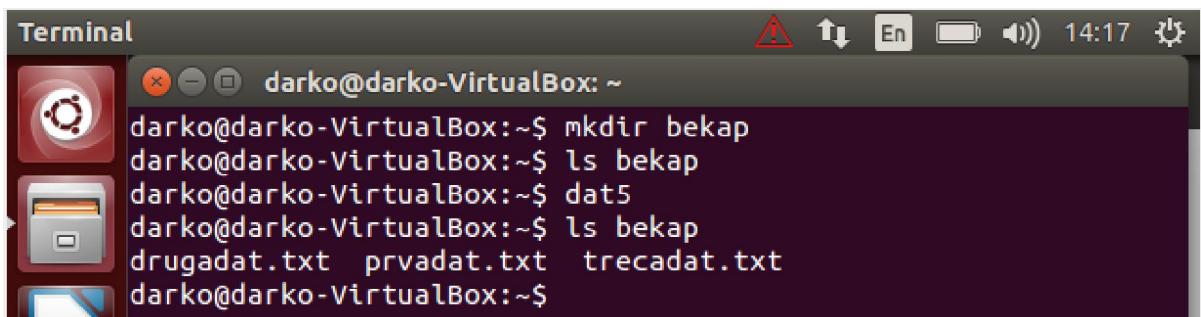
A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark background and contains the following text:

```
darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ cat dat5
#!/bin/bash

# sve datoteke koje se zavrsavaju na .txt premjestaju se u bekap
# uslov je da datoteka nije prazna

for dat in *.txt
do
[ -s $dat ] && mv $dat bekap
done
darko@darko-VirtualBox:~$
```

Slika 4.13.1. – Kod skripta dat5 koji premešta datoteke koje nisu prazne i koje se završavaju na .txt u direktorijum bekap



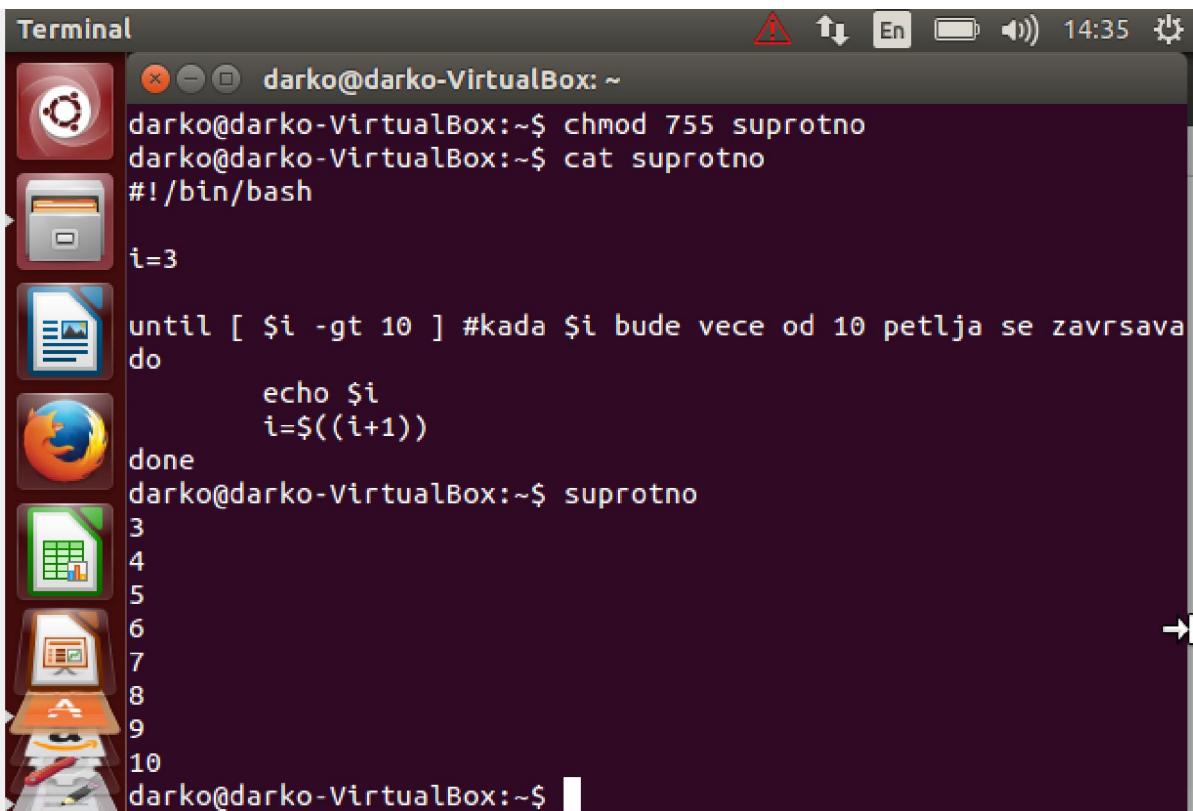
A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark background and contains the following text:

```
darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ mkdir bekap
darko@darko-VirtualBox:~$ ls bekap
darko@darko-VirtualBox:~$ dat5
darko@darko-VirtualBox:~$ ls bekap
drugadat.txt prvadat.txt trecadat.txt
darko@darko-VirtualBox:~$
```

Slika 4.13.2. – Testiranje skripta dat5

4.14. until

Ova petlja ima isti oblik kao i while. Suštinska razlika je u tome sto se until izvršava sve dok je izlazni status komande u uslovu različit od nula.

A screenshot of a Linux desktop environment, specifically Ubuntu, showing a terminal window titled "Terminal". The terminal window contains a shell script named "suprotno" which uses a "until" loop to print numbers from 3 to 10. The script starts with "#!/bin/bash", initializes \$i=3, and then enters a loop where it prints the value of \$i, increments \$i by 1, and checks if \$i is greater than 10. Once \$i reaches 10, the loop exits and the script ends. The terminal window also shows the user's name "darko" and the host "darko-VirtualBox".

```
darko@darko-VirtualBox:~$ chmod 755 suprotno
darko@darko-VirtualBox:~$ cat suprotno
#!/bin/bash

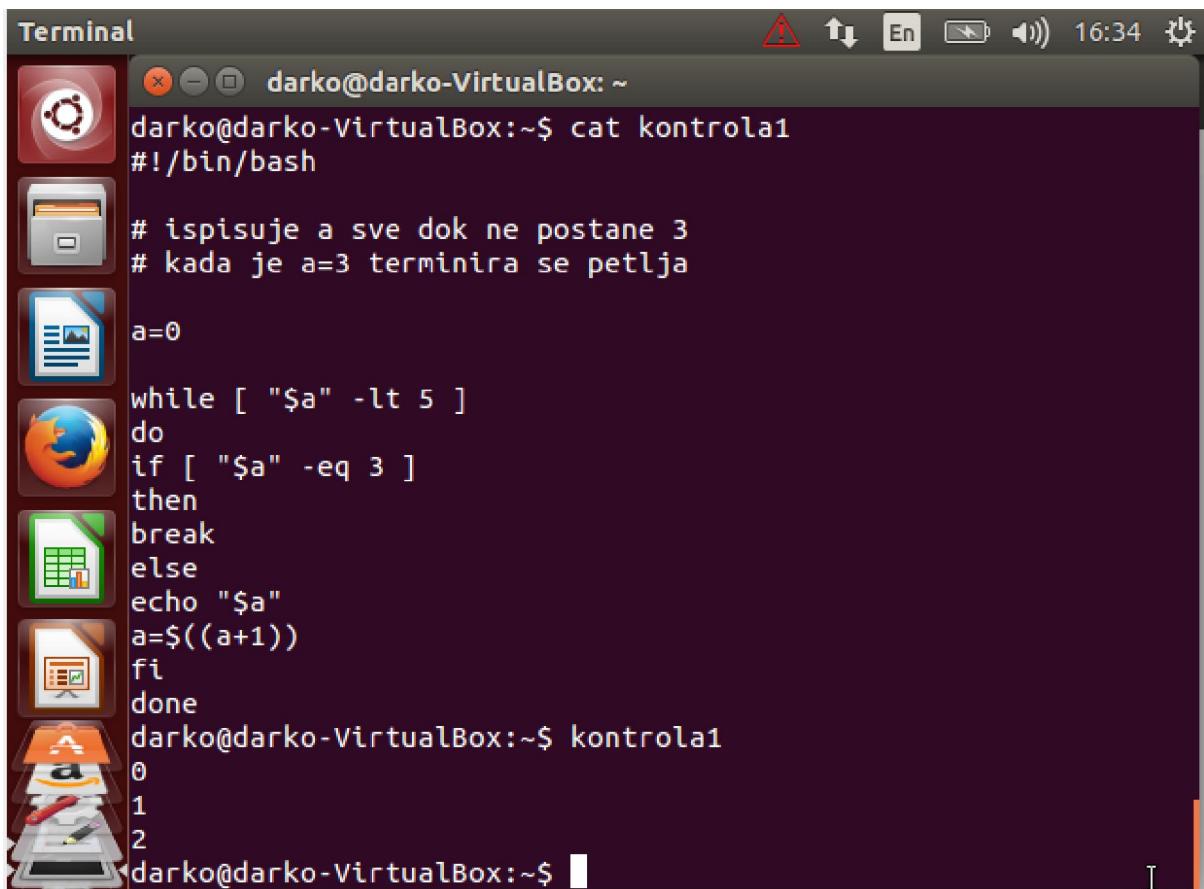
i=3

until [ $i -gt 10 ] #kada $i bude vece od 10 petlja se zavrsava
do
    echo $i
    i=$((i+1))
done
darko@darko-VirtualBox:~$ suprotno
3
4
5
6
7
8
9
10
darko@darko-VirtualBox:~$
```

Slika 4.14.1. – Kod skripta koji ispisuje brojne vrijednosti koristeći until petlju

4.15. break i continue

Ove naredbe se koriste za kontrolu toka petlje. Naredba break trenutno terminira petlju. Naredba continue prekida tekuću iteraciju i prelazi na sledeću.

A screenshot of an Ubuntu desktop environment. On the left, there's a dock with various icons: Dash, Home, Applications, System, Help, and a few others. A terminal window is open in the center. The title bar says "Terminal". The window contains the following text:

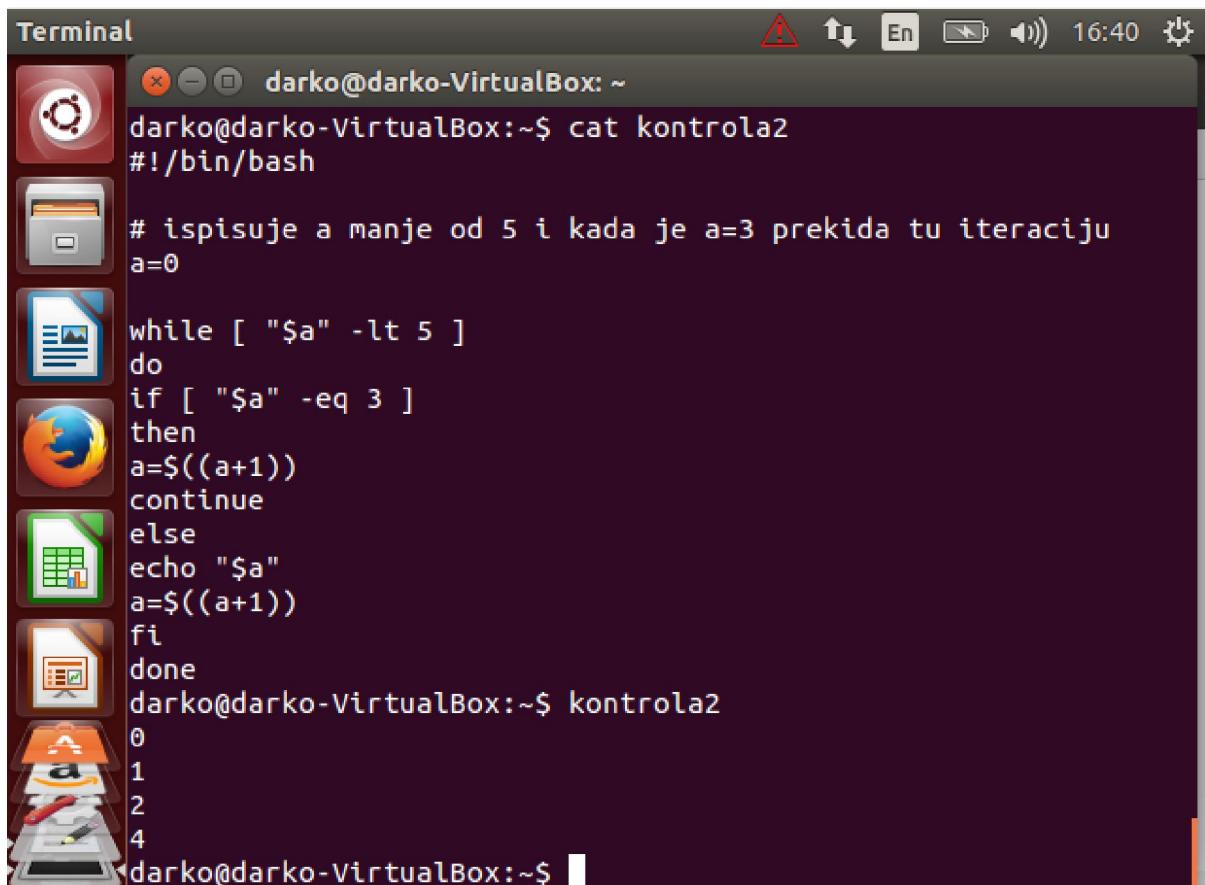
```
darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ cat kontrola1
#!/bin/bash

# ispisuje a sve dok ne postane 3
# kada je a=3 terminira se petlja

a=0

while [ "$a" -lt 5 ]
do
if [ "$a" -eq 3 ]
then
break
else
echo "$a"
a=$((a+1))
fi
done
darko@darko-VirtualBox:~$ kontrola1
0
1
2
3
```

Slika 4.15.1. – Kod skripta koji ilustruje primjenu naredbe break



```
darko@darko-VirtualBox: ~
darko@darko-VirtualBox:~$ cat kontrola2
#!/bin/bash

# ispisuje a manje od 5 i kada je a=3 prekida tu iteraciju
a=0

while [ "$a" -lt 5 ]
do
if [ "$a" -eq 3 ]
then
a=$((a+1))
continue
else
echo "$a"
a=$((a+1))
fi
done
darko@darko-VirtualBox:~$ kontrola2
0
1
2
3
4
darko@darko-VirtualBox:~$
```

Slika 4.15.2. – Kod skripta koji ilustruje primjenu naredbe continue

4.16. Funkcije

4.16.1. Definiranje i pozivanje funkcija

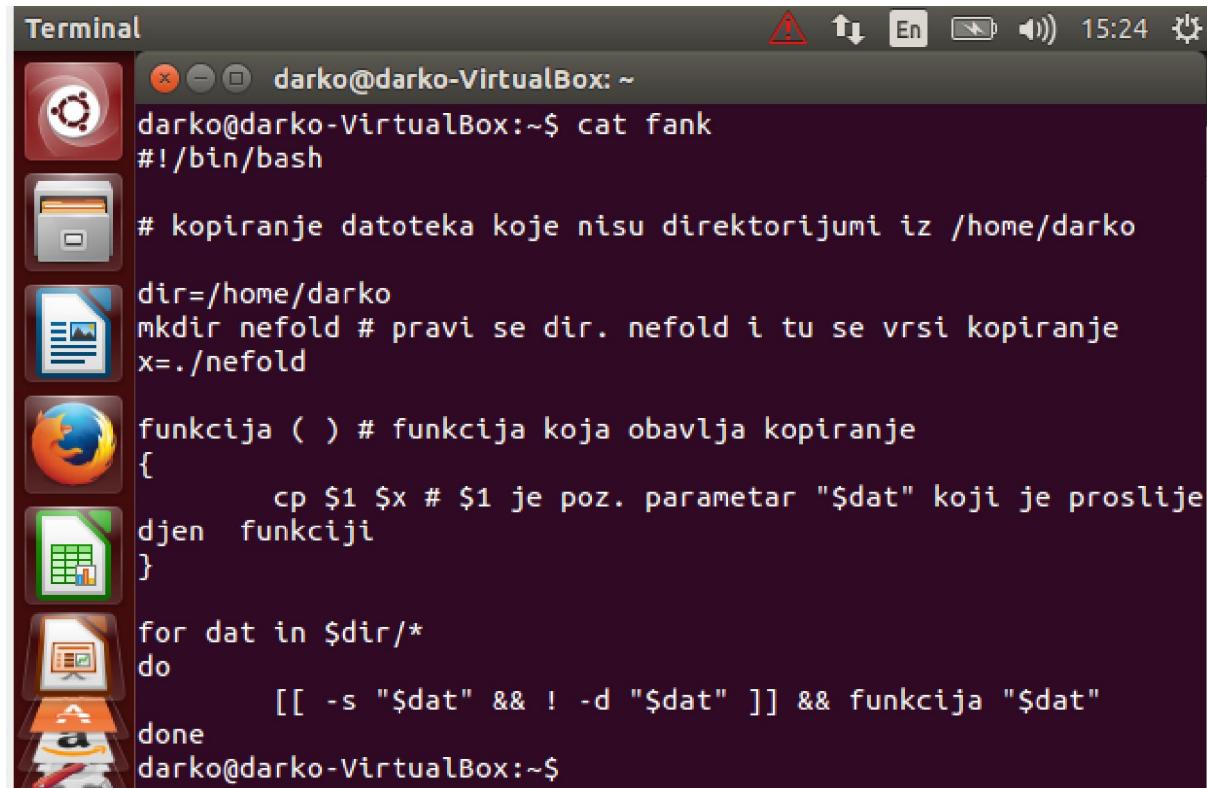
Funkcija predstavlja blok naredbi koji se može izvršiti iz bilo kog dijela programa jednostavno navodeći njeno ime. Ima sledeću strukturu:

```
ime_funkcije ()
{
    blok naredbi
}
```

Ova funkcija se poziva unošenjem *ime_funkcije*. Pravila za davanje imena funkcijama su ista kao i pravila za davanje imena promjenljivima.

4.16.2. Pozicioni parametri

Ljuska ima niz promjenljivih zvanih pozicioni parametri, koje sadrže pojedinačne riječi na komandnoj liniji [1]. Označavaju se brojevima od 0 do 9. Za pozicione parametre iznad 9 koriste se vitičaste zagrade. Promjenljiva \$# označava ukupan broj pozicionih parametara.



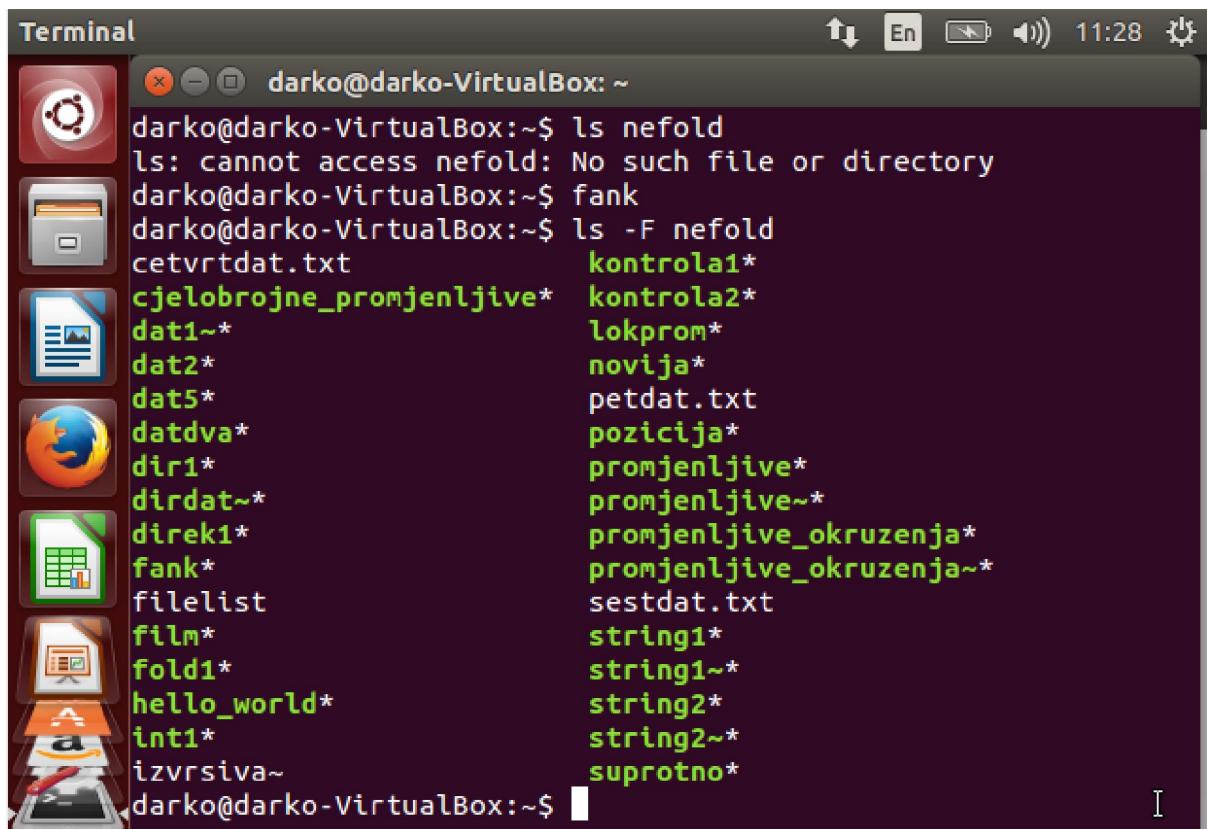
```
darko@darko-VirtualBox:~$ cat fank
#!/bin/bash

# kopiranje datoteka koje nisu direktorijumi iz /home/darko
dir=/home/darko
mkdir nefold # pravi se dir. nefold i tu se vrši kopiranje
x=./nefold

funkcija ( ) # funkcija koja obavlja kopiranje
{
    cp $1 $x # $1 je poz. parametar "$dat" koji je proslije
djen funkciji
}

for dat in $dir/*
do
    [[ -s "$dat" && ! -d "$dat" ]] && funkcija "$dat"
done
darko@darko-VirtualBox:~$
```

Slika 4.16.2.1. – Kod skripta fank. Definisana je funkcija koja vrši željeno kopiranje, nakon čega sledi for petlja u glavnom programu. Funkcija mora biti definisana prije glavnog programa.



The screenshot shows a terminal window titled "Terminal" running on an Ubuntu desktop. The window contains the following command-line session:

```
darko@darko-VirtualBox:~$ ls nefold
ls: cannot access nefold: No such file or directory
darko@darko-VirtualBox:~$ fank
darko@darko-VirtualBox:~$ ls -F nefold
cetvrtdat.txt          kontrola1*
cjenlobrojne_promjenljive* kontrola2*
dat1~*                  lokprom*
dat2*                   novija*
dat5*                   petdat.txt
datdva*                 pozicija*
dir1*                   promjenljive*
dirdat~*                promjenljive~
direk1*                 promjenljive_okruzenja*
fank*                   promjenljive_okruzenja~
filelist                sestdat.txt
film*                   string1*
fold1*                  string1~*
hello_world*             string2*
int1*                   string2~*
izvrsiva~               suprotno*
```

Slika 4.16.2.2. – Pozivanje skripta fank. Skript je kreirao direktorijum nefold i u njega kopirao sve datoteke koje nisu prazne i koje nisu direktorijumi.

```
komfank (~) - gedit
komfank x
#!/bin/bash

echo "Unesite dvije datoteke:"
read dat1 dat2

komparacija ()
{
    echo "Putanja ovog programa je: $0"
    echo "Prvi pozicioni parametar je: $1"
    echo "Drugi pozicioni parametar je: $2"

    if [[ "$1" -nt "$2" ]]
    then
        echo "Datoteka $1 je novija od datoteke $2."
    else
        echo "Datoteka $2 je novija od datoteke $1."
    fi
}
```

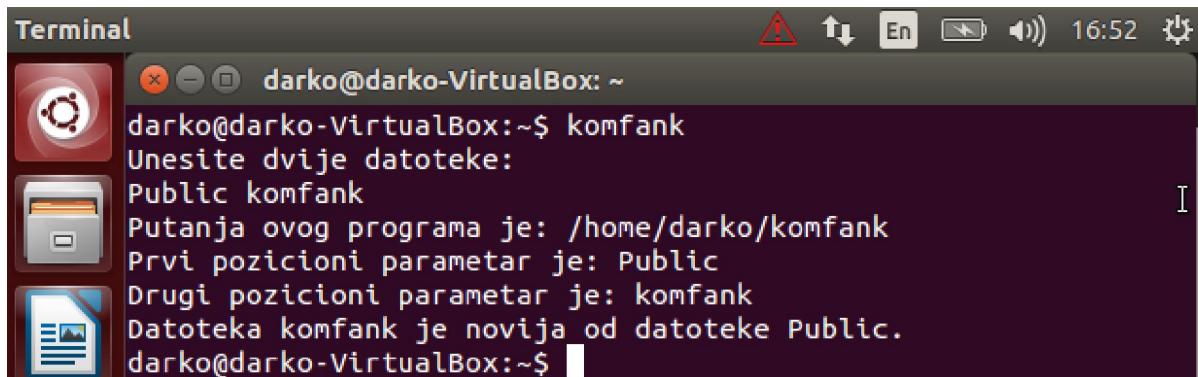
Slika 4.16.2.3. – Prvi dio skripta komfank. Od korisnika se traži da unese imena dviju datoteke. Definisana je funkcija komparacija koja ispituje koja je od unesenih datoteka novija. Ova funkcija ispisuje \$0 parametar koji označava putanju skripta komfank. Takođe, ispisuje i dva parametra \$1 i \$2. Ovi parametri su imena dviju datoteka koje su proslijedene funkciji komparacija na obradu.

```
echo "Datoteka $1 je novija od datoteke $2."
else
    echo "Datoteka $2 je novija od datoteke $1."
fi

}

if [[ -e "$dat1" && -e "$dat2" && "$dat1" != "$dat2" ]]
then
    komparacija "$dat1" "$dat2"
else
    echo "Pogresan unos."
fi
```

Slika 4.16.2.4. – Drugi dio skripta komfank. U glavnom programu se ispituje da li dati unosi predstavljaju datoteke i pritom moraju da budu različiti da bi se komparacija uspješno izvršila. Ako je testiranje uspješno, poziva se funkcija sa argumentima koji su unijeti sa tastature. Ako nije, ispisuje se poruka o pogrešnom unosu.



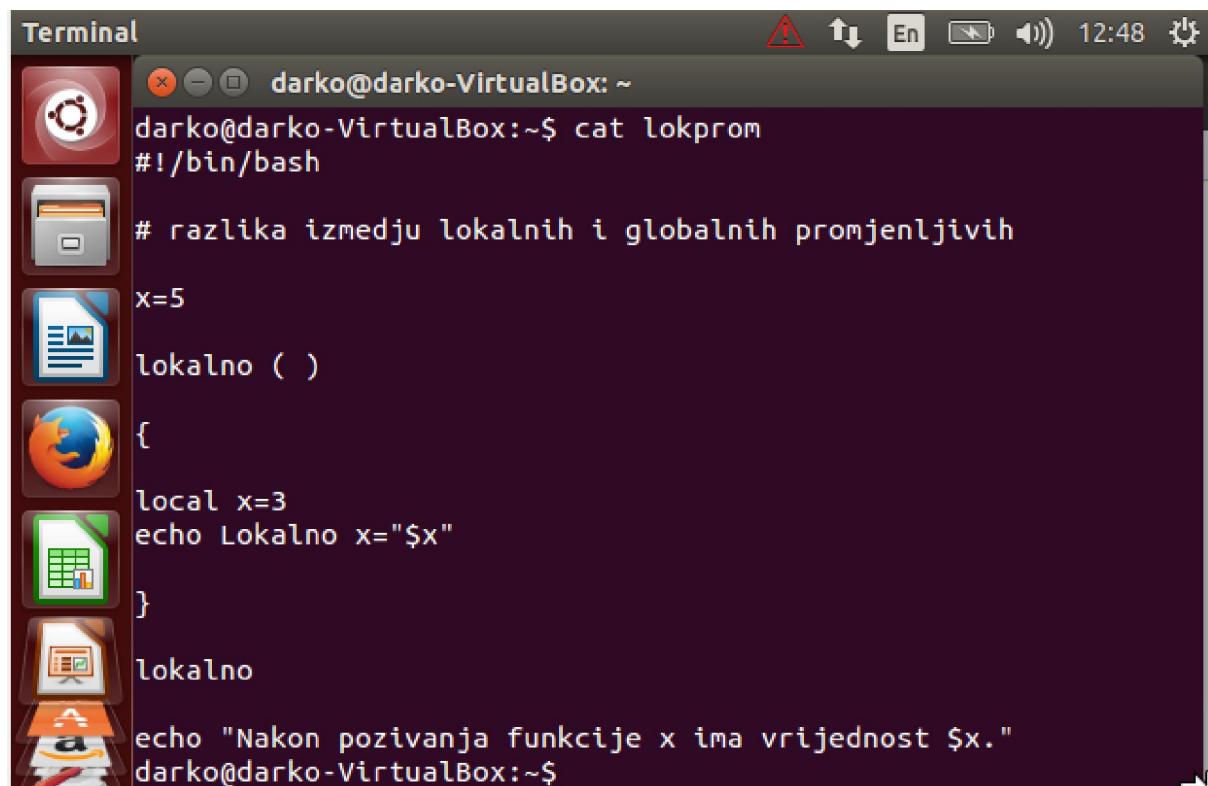
A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark background and contains the following text:

```
darko@darko-VirtualBox: ~$ komfank
Unesite dvije datoteke:
Public komfank
Putanja ovog programa je: /home/darko/komfank
Prvi pozicioni parametar je: Public
Drugi pozicioni parametar je: komfank
Datoteka komfank je novija od datoteke Public.
darko@darko-VirtualBox:~$
```

Slika 4.16.2.5. - Testiranje skripta komfank

4.16.3. Lokalne promjenljive

To su promjenljive koje važe samo u funkcijama. Ako se u funkciji želi napraviti promjenljiva x to se postiže unošenjem local x . Ova promjenljiva x će važiti samo u datoј funkciji. Ako u glavnom programu postoji globalna promjenljiva x njenu vrijednost neće biti moguće promijeniti korišćenjem lokalne promjenljive x .



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark background and contains the following text:

```
darko@darko-VirtualBox: ~$ cat lokprom
#!/bin/bash

# razlika izmedju lokalnih i globalnih promjenljivih

x=5

lokalno ( )

{

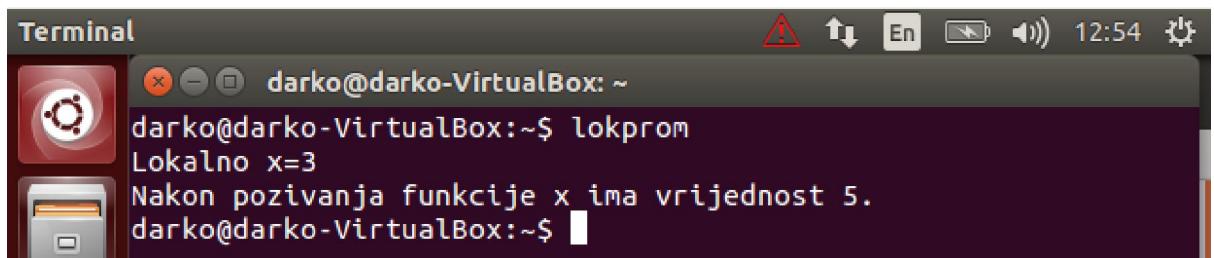
local x=3
echo Lokalno x="$x"

}

lokalno

echo "Nakon pozivanja funkcije x ima vrijednost $x."
darko@darko-VirtualBox:~$
```

Slika 4.16.3.1. – Kod skripta lokprom. Ilustruje razliku izmedu lokalnih i globalnih promjenljivih.



The screenshot shows a terminal window titled "Terminal". The window has a dark theme with a red header bar. The title bar displays the terminal icon, the title "Terminal", and the user information "darko@darko-VirtualBox: ~". The main area of the terminal shows the following text:
darko@darko-VirtualBox:~\$ lokprom
Lokalno x=3
Nakon pozivanja funkcije x ima vrijednost 5.
darko@darko-VirtualBox:~\$ █

Slika 4.16.3.2. – Izvršavanje skripta lokprom

5.ZAKLJUČAK

Najveća prednost pisanja komandnih skriptova leži u tome je to što je sintaksa ista kao da se komande unose sa komandne linije [4]. Često je mnogo brže napisati skript, nego kod sa identičnim učinkom, u nekom drugom programskom jeziku. Dobra strana je što komandni skriptovi omogućavaju automatizovanje operacija sa datotekama, koje bi se inače ručno obavljale.

Mnoge komande kao npr. cd i cp se razlikuju u samo jednom slovu pa pogrešan unos može prouzrokovati probleme. Zloupotreba znaka > može obrisati neku važnu datoteku u sistemu. Unošenjem *>datoteka* sadržaj datoteke *datoteka* će biti u potpunosti obrisan. Komandni skriptovi nisu namijenjeni za složene aritmetičke proračune. Ovi skriptovi su predviđeni da budu kratki. Pisanje dugačkih skriptova nije nešto što je poželjno.

LITERATURA

- [1] Linux s komandne linije, William E. Shotts, Mikro knjiga
- [2] Kako radi Linux, Brian Ward, Mikro knjiga
- [3] Advanced Bash-Scripting Guide, Mendel Cooper
- [4] https://en.wikipedia.org/wiki/Shell_script#Other_scripting_languages