

Uvod u Linux

D105



Ovu inačicu priručnika izradio je autorski tim Srca u sastavu:

Autor: Vladimir Braus

Urednik: Suzana Kikić

Lektor: Jasna Novak Milić

TEČAJEVI**srca**

Sveučilište u Zagrebu

Sveučilišni računski centar

Josipa Marohnića 5, 10000 Zagreb

tecajevi@srce.hr

ISBN: 978-953-7138-44-8 (meki uvez)

ISBN: 978-953-7138-45-5 (PDF)

Verzija priručnika D105-20141205



Ovo djelo dano je na korištenje pod licencom Creative Commons
Imenovanje-Nekomercijalno-Dijeli pod istim uvjetima 4.0 međunarodna.
Licenca je dostupna na stranici:
<http://creativecommons.org/licenses/by-nc-sa/4.0/>.

Sadržaj

Uvod	1
1. Osnovni pojmovi	3
1.1. Osnove rada u <i>Linuxu</i>	3
1.2. Načini rada	3
1.3. Udaljeni pristup	3
1.4. Vježba: Prijavljivanje na sustav i osnove rada s programom PuTTY	4
1.5. Pitanja za ponavljanje	6
2. Naredbe u <i>Linuxu</i>	7
2.1. Radni direktorij	7
2.2. Prompt	7
2.3. Izgled naredbi	8
2.4. Promjena lozinke	8
2.5. Odjavljivanje sa sustava	9
2.6. Naredbe <code>echo</code> i <code>clear</code>	9
2.7. Informacije o korisnicima na sustavu: <code>who</code> , <code>w</code> , <code>finger</code> i <code>last</code>	9
2.8. Komunikacija s drugim korisnicima: <code>write</code> i <code>mesg</code>	11
2.9. Sustav pomoći: <code>man</code> i <code>info</code>	11
2.10. Vježba: Unos naredbi, komunikacija s drugim korisnicima i završetak rada na sustavu	12
2.11. Pitanja za ponavljanje	14
3. Sustav datoteka	15
3.1. Općenito o sustavu datoteka	15
3.2. Imena datoteka	15
3.3. Ispis radnog direktorija: <code>pwd</code>	16
3.4. Apsolutne i relativne staze	17
3.5. Promjena radnog direktorija: <code>cd</code>	18
3.6. Ispis sadržaja direktorija: <code>ls</code>	18
3.7. Stvaranje i uklanjanje direktorija: <code>mkdir</code> i <code>rmdir</code>	19
3.8. Stvaranje datoteka: <code>touch</code>	20
3.9. Brisanje datoteka: <code>rm</code>	20
3.10. Kopiranje i premještanje datoteka i direktorija: <code>cp</code> i <code>mv</code>	21
3.11. Ispis sadržaja datoteka: <code>cat</code> , <code>more</code> i <code>less</code>	22
3.12. Zamjenski znakovi	22
3.13. Pretraživanje sustava datoteka: <code>find</code>	23
3.14. (π) Vježba: Rad s datotekama i direktorijima	23
3.15. Pitanja za ponavljanje	24
4. Prava pristupa datotekama i direktorijima	25
4.1. Naredba <code>id</code>	25
4.2. Razine prava pristupa: <i>user</i> , <i>group</i> i <i>other</i>	25
4.3. Vrste prava pristupa: <i>read</i> , <i>write</i> i <i>execute</i>	26
4.4. Postavljanje prava pristupa pomoću simboličkih oznaka	27

4.5. Postavljanje prava pristupa pomoću numeričkih oznaka.....	28
4.6. Naredba <code>umask</code>	29
4.7. Vježba: postavljanje prava pristupa	30
4.8. Pitanja za ponavljanje	31
5. Uređivač teksta <i>GNU nano</i>	32
5.1. Općenito o uređivačima teksta.....	32
5.2. Osnovna svojstva uređivača teksta <i>GNU nano</i>	32
5.3. Pokretanje programa.....	32
5.4. Osnovne komande	34
5.6. Vježba: Rad s uređivačem teksta <i>GNU nano</i>	35
5.7. Pitanja za ponavljanje	36
6. Ljuska <i>Bash</i>.....	37
6.1. Što je ljuska?	37
6.2. Upis naredbi	37
6.3. Varijable	37
6.3. Lokalne varijable	38
6.4. Varijable okruženja i naredba <code>export</code>	39
6.5. Vrste naredbi.....	39
6.6. Vježba: Varijable i naredbe	41
6.7. Pitanja za ponavljanje	42
7. Preusmjeravanje ulaza i izlaza naredbi	44
7.1. <code>stdin</code> , <code>stdout</code> i <code>stderr</code>	44
7.2. Filtri.....	45
7.3. Preusmjeravanje ulaza: <code><</code>	45
7.4. Preusmjeravanje izlaza: <code>></code> i <code>>></code>	48
7.5. Preusmjeravanje standardnog izlaza za poruke o greškama: <code>2></code> i <code>2>></code>	48
7.6. Ulančavanje	49
7.7. Vježba: Preusmjeravanje ulaza i izlaza, filtri i ulančavanje naredbi.....	50
7.8. Pitanja za ponavljanje	50
8. Procesi	51
8.1. Općenito o procesima	51
8.1. Popis procesa: <code>ps</code> i <code>top</code>	51
8.2. Izvršavanje naredbi u pozadini.....	52
8.4. Upravljanje procesima u ljusci	52
8.3. Signali i naredba <code>kill</code>	53
8.4. Vježba: Upravljanje procesima.....	54
8.5. Pitanja za ponavljanje	55
9. Arhiviranje i sažimanje datoteka.....	56
9.1. Arhiviranje datoteka: <code>tar</code>	56
9.2. Sažimanje datoteka: <code>gzip</code>	57
9.3. Vježba: Arhiviranje i sažimanje datoteka	58
9.4. Pitanja za ponavljanje	58

10. <i>Linux</i> na mreži.....	59
10.1. Pristup udaljenim sustavima: <i>ssh</i> i <i>scp</i>	59
10.2. Mrežno ime i domena sustava: <i>hostname</i> i <i>dnsdomainname</i>	59
10.3. Prevođenje mrežnih adresa: <i>nslookup</i> , <i>host</i> i <i>dig</i>	60
10.4. Provjera dostupnosti udaljenih sustava: <i>ping</i> i <i>traceroute</i>	61
10.5. Vježba: <i>Linux</i> na mreži	62
10.6. Pitanja za ponavljanje.....	63
 A. Dodatak: Mrežne postavke sustava <i>Linux</i>	64
A.1. Preduvjeti za rad na mreži	64
A.2. Mrežna adresa računala i izlazni usmjerivač (<i>gateway</i>)	64
A.3. Simboličko ime računala i DNS-poslužitelji	66

Uvod

Tečaj **Uvod u *Linux* – rad s naredbama (D105)** obrađuje osnove rada u tekstualnom okruženju sustava *Linux*. Namijenjen je polaznicima koji imaju malo iskustva u radu s *Linuxom* ili do sada uopće nisu radili na *Linuxu*.

Za pohađanje ovog tečaja dovoljno je sljedeće predznanje: poznavanje osnova rada s računalom i operacijskim sustavom *MS Windows* te poznavanje osnova rada na Internetu.

U ovom su priručniku imena i pojedini elementi naredbi, imena datoteka i direktorija, korisnička imena te imena varijabli pisani proporcionalnim slovima (na primjer: naredba `passwd`, datoteka `/tmp/test`, korisnik `tecaj01`, varijabla `PS1`).

Sintaksa naredbi pisana je podebljanim proporcionalnim slovima. Na primjer:

```
cat [DATOTEKA...]
tar x[v]f ARHIVA [POPIS]
msg [y|n]
```

Pri tome vrijedi:

- objekti koji nisu pisani u uglatim zagradama su obavezni elementi sintakse
- objekte koji su u zagradama može se prilikom pisanja naredbi ispustiti (nisu obavezni)
- objekti pisani malim slovima navedeni su eksplicitno (treba ih pisati točno onako kako je navedeno u sintaksi, obično su to opcije i nazivi naredbi)
- objekti pisani velikim slovima su promjenjiva imena (obično su to argumenti naredbi)
- tri točkice (. . .) označavaju da se prethodni element može ponavljati proizvoljan broj puta
- znak | (*pipe*) označava da na navedenom mjestu treba pisati jedan od susjednih objekata.

Primjeri su pisani proporcionalnim slovima. Poruke o greškama pisane su crveno. Na primjer:

```
$ ls -a
.profile doc test
$ ls abc
ls: cannot access abc: No such file or directory
$
```

Tipke i kombinacije tipki (kontrolni znakovi) pisane su proporcionalnim slovima u uglatim zagradama (na primjer: `[Backspace]`, `[B]`, `[Ctrl-C]`, `[Alt-X]`).

Važni su pojmovi prilikom prvog spominjanja pisani podebljano.

Strani izrazi i nazivi pisani su kurzivom.

1. Osnovni pojmovi

1.1. Osnove rada u Linuxu

Linux omogućava istovremeni rad više korisnika, stoga kažemo da je *Linux* **više-korisnički sustav**.

Svakom korisniku dodijeljen je **korisnički račun** – skup resursa koje može koristiti. Primjeri resursa koje korisnici mogu koristiti su diskovni prostor, procesorsko vrijeme, radna memorija, pisači i drugo.

Kako bi sustav mogao razlikovati korisničke račune (a time i same korisnike na sustavu), svaki korisnički račun ima svoje **ime**. Taj je podatak, zajedno s korisničkom **lozinkom** potrebno upisati prilikom prijavljivanja za rad na sustavu. Na taj način sustav zna koji se korisnik prijavljuje i provjerava njegov identitet. Korisničko je ime javni podatak, dok je lozinka tajni podatak.

Sustav se brine o tome da su korisnička radna okruženja međusobno odvojena te da korisnici mogu upravljati svojim radnim okruženjem – određivati prava pristupa do svojih podataka, podešavati svoje radno okruženje (putem konfiguracijskih datoteka), pokretati programe i upravljati njihovim izvođenjem i drugo.

Napomena

Korisničko ime administratora sustava je **root**.

Napomena

Važno je čuvati tajnost lozinki!

1.2. Načini rada

Linux omogućava više načina rada. Prvi način rada je neposredan rad na konzoli ili terminalu koji su direktno spojeni na računalo. Drugi je način udaljen – putem računalne mreže i drugog računala koje simulira terminal.

U oba je slučaja moguće raditi u tekstualnom ili grafičkom radnom okruženju.

U **tekstualnom radnom okruženju** sustavom se upravlja unosom tekstualnih naredbi.

U **grafičkom radnom okruženju** sustavom se upravlja putem grafičkog sučelja u kojem su pojedini elementi sustava i naredbe prikazani kao grafički objekti (ikone).

Napomene

Terminali se sastoje od ekrana i tipkovnice.

Konzola je terminal putem kojeg se upravlja računalom tijekom pokretanja sustava te se na njega ispisuju kritične poruke za vrijeme rada sustava.

Napomena

Rad u grafičkom radnom okruženju sustava *Linux*, napredno služenje sustavom *Linux* te sistemska administracija sustava *Linux* nisu predmet ovog tečaja.

1.3. Udaljeni pristup

Komunikaciju između udaljenog i lokalnog sustava određuje odabrani protokol – jezik kojim se služe računala za povezivanje i razmjenu podataka.

Danas se za terminalski pristup udaljenim računalima najčešće koristi protokol **SSH** koji omogućava sigurno povezivanje računala kriptiranjem podataka koji se razmjenjuju među računalima.

Postoje i drugi protokoli za terminalski pristup udaljenim računalima. Neki od njih (na primjer, **telnet** ili **rlogin**) ne koriste kriptiranje prilikom razmjene

Napomena

Nesigurne protokole koji ne koriste kriptiranje prilikom razmjene podataka treba izbjegavati!

podataka. Ti se protokoli smatraju nesigurnima i njihovo korištenje treba izbjegavati.

Da bi se putem lokalnog računala moglo raditi na udaljenom računalu, potrebno je na lokalnom računalu imati instaliran odgovarajući program (klijent). Taj program ima dvojni funkciju: on ostvaruje vezu s udaljenim računalom (rabeći pri tome odgovarajući protokol) te simulira rad terminala.

PuTTY je jedan od takvih programa. Besplatan je i moguće ga je preuzeti na adresi: <http://www.chiark.greenend.org.uk/~sgtatham/putty/>.

Instalacija tog programa je vrlo jednostavna: nakon preuzimanja s Interneta dovoljno je program raspakirati i kopirati na željeno mjesto na disku.

Da bi se mogla uspostaviti veza s udaljenim računalom, potrebno je poznavati njegovo mrežno (simboličko) ime ili njegovu mrežnu (numeričku) adresu.

Mrežna imena strojeva su oblika *imestroja.domena* (na primjer, baltazar.srce.hr), a numeričke adrese su nizovi od četiri okteta razdvojenih točkom (na primjer, 161.53.2.82).

1.4. Vježba: Prijavljivanje na sustav i osnove rada s programom PuTTY

Da bi se ova vježba mogla izvesti potrebno je:

- na lokalnom stroju imati instaliran program PuTTY
- poznavati mrežno ime stroja na kojem će se raditi
- imati korisnički račun na udaljenom stroju (poznavati svoje korisničko ime i lozinku).

Navedene podatke dobit ćete od predavača na početku vježbe. Zapamtite ili zabilježite te podatke jer ćete ih često koristiti tijekom ovog tečaja.

1. Pokrenite program PuTTY.



PuTTY

Program PuTTY pokreće se na neki od uobičajenih načina: dvostrukim klikom na ikonu koja na radnoj površini ili na tvrdom disku predstavlja navedeni program ili odgovarajućim odabirom u izborniku sustava.

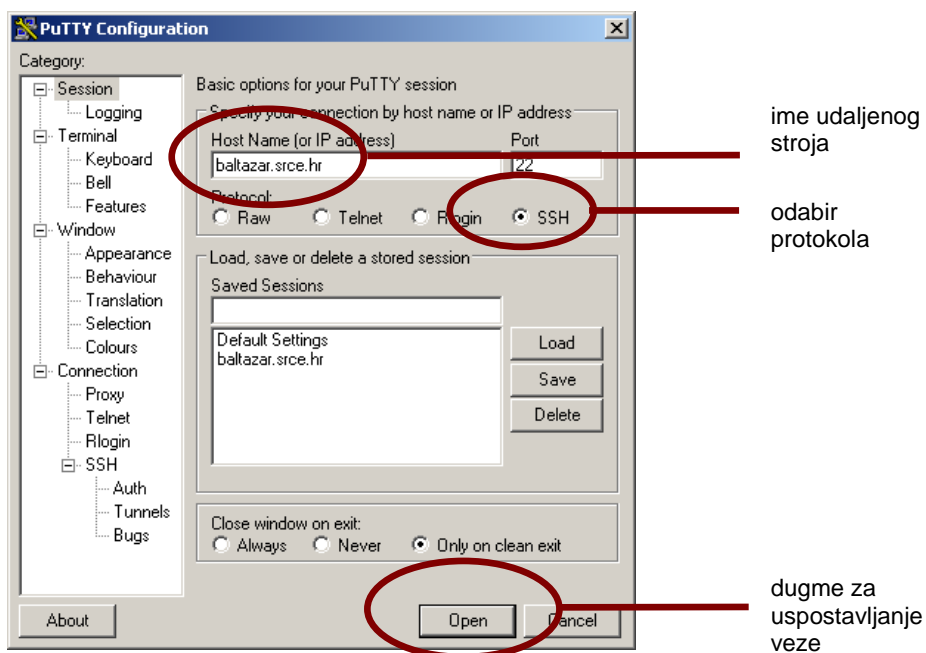
2. Odaberite i upišite osnovne parametre veze.

Nakon pokretanja programa, pojavit će se prozor u kojem je moguće odabrati osnovne postavke veze koja se želi uspostaviti.

Upišite ime udaljenog stroja i odaberite protokol SSH (kao na slici). Nakon toga kliknite na dugme **Open**.

Program će nakon toga otvoriti novi prozor i uspostaviti vezu s udaljenim računalom.

Napomena: Prilikom prvog uspostavljanja veze s nekim računalom, PuTTY od korisnika traži odobrenje da uspostavi odnos povjerenja s navedenim računalom. Da biste programu potvrdili da može nastaviti s radom, dovoljno je da kliknete na dugme **Ok**.



3. Prijavite se na sustav.

Nakon uspostavljanja veze s udaljenim računalom, potrebno je prijaviti se za rad na sustavu. Upišite svoje korisničko ime i lozinku.

```

baltazar.srce.hr - PuTTY
login as: tecaj01
tecaj01@baltazar.srce.hr's password:

```

Nakon uspješne prijave, sustav je ispisao nekoliko redaka teksta. Što možete zaključiti o sustavu na osnovi teksta koji je ispisao?

4. Promijenite vrstu i veličinu slova koja se koriste za prikaz teksta u prozoru programa PuTTY.

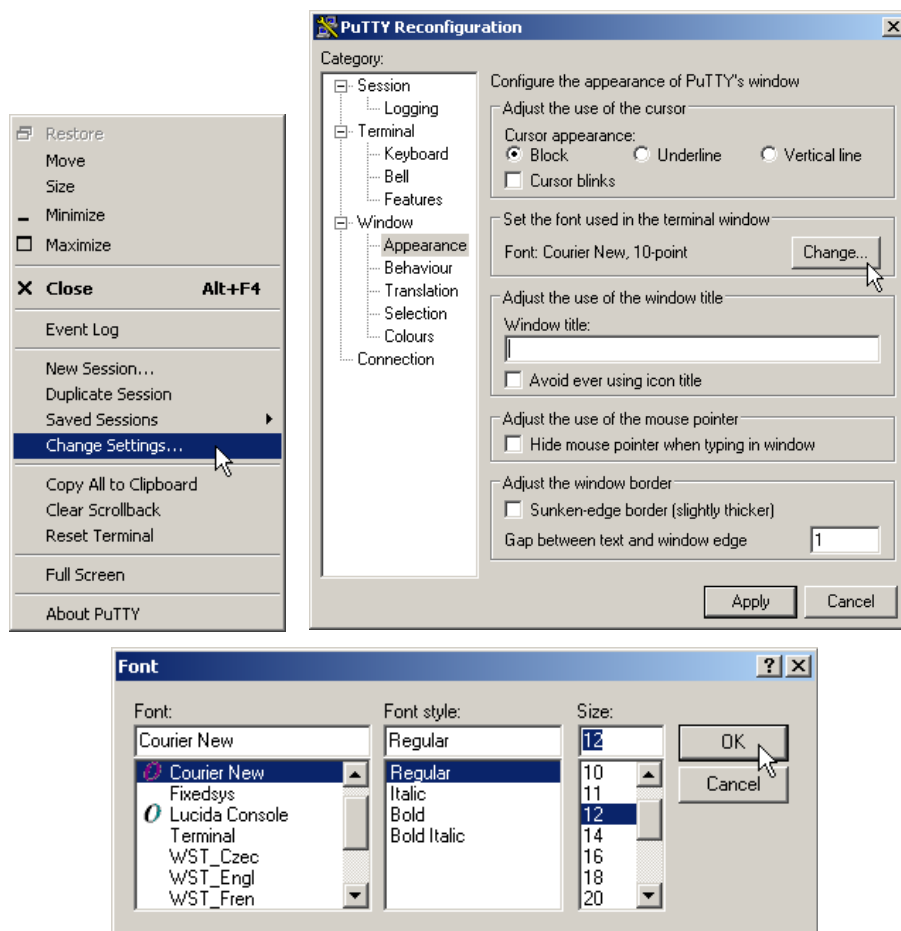
Kliknite desnom tipkom miša na gornji dio okvira radnog prozora programa PuTTY. Pojavit će se izbornik.

Odaberite **Change Settings**. Pojavit će se novi prozor **PuTTY Reconfiguration**.

U novom prozoru u okviru **Category** odaberite **Window/Appearance** i unutar okvira s naslovom **Set the font used in terminal window** kliknite na dugme **Change...**

U novom prozoru odaberite sljedeći font: **Courier New, Regular, 12 pt** i kliknite na **Ok**.

U prozoru **PuTTY Reconfiguration** kliknite na dugme **Apply** koje se nalazi na dnu prozora.



U novom prozoru odaberite sljedeći font: **Courier New, Regular, 12 pt** i kliknite na **Ok**.

U prozoru **PuTTY Reconfiguration** kliknite na dugme **Apply** koje se nalazi na dnu prozora.

1.5. Pitanja za ponavljanje

1. *Linux* je višekorisnički sustav. Što to znači?
2. Što je korisnički račun?
3. Što je potrebno upisati prilikom prijavljivanja na sustav?
4. Na koje se načine može raditi na *Linuxu*?
5. Po čemu se razlikuje tekstualno od grafičkog radnog okruženja?
6. Što su protokoli i čemu služe?
7. Koji se protokol danas najčešće koristi za rad na udaljenim sustavima i zašto?
8. Koje je podatke nužno poznavati da bi se moglo raditi na nekom udaljenom sustavu?
9. Ako je prijava na sustav neuspješna, koji su mogući razlozi za to?

2. Naredbe u *Linuxu*

2.1. Radni direktorij

Tijekom rada u *Linuxu* uvijek se nalazimo u nekom direktoriju (mapi). To znači da kada *Linuxu* zadajemo neku naredbu i ako nismo drugačije naveli, ona se odnosi upravo na taj direktorij i njegov sadržaj.

Taj se direktorij zove **radni direktorij**.

Na primjer, ako *Linuxu* zadamo da stvori neku datoteku, on će je stvoriti upravo u radnom direktoriju (osim ako nismo izravno naveli da to napravi u nekom drugom direktoriju).

Prvi radni direktorij u kojem se nađemo nakon prijave na sustav zove se **početni direktorij** (*home*).

Svaki korisnik ima svoj početni direktorij.

Više o direktorijima objašnjeno je u poglavlju 4. *Sustav datoteka*.

2.2. Prompt

Prompt je niz znakova koje sustav ispisuje kao obavijest da je spreman za izvršenje sljedeće naredbe.

Uobičajeno se prompt sastoji od korisničkog imena, imena stroja i(li) imena radnog direktorija međusobno odijeljenih znakovima (separatorima) @, : i \$.

Na primjer:

```
tecaj01@baltazar.srce.hr:/tmp$
```

- `tecaj01` je korisničko ime
- `@` je znak kojim se razdvaja korisničko ime od ostatka prompta (prvi separator)
- `baltazar.srce.hr` je ime stroja
- `:` je znak kojim se razdvaja ime stroja od ostatka prompta (drugi separator)
- `/tmp` je ime radnog direktorija
- `$` je znak koji označava kraj prompta.

Izgled prompta moguće je mijenjati. Izmjene mogu raditi sistemski administratori (za sve korisnike na sustavu) i sami korisnici (za svoje potrebe).

Primjer vrlo jednostavnog prompta je: `$`.

Početni direktorij se često označava znakom `~` (tilda).

Na primjer, ako je naš početni direktorij `/home/tecaj01` i ako se nalazimo u njemu, tada će naš prompt umjesto

```
tecaj01@baltazar.srce.hr:/home/tecaj01$
```

izgledati ovako:

```
tecaj01@baltazar.srce.hr:~$ .
```

Napomena

Znak `~` (tilda) označava početni direktorij (*home*).

2.3. Izgled naredbi

Naredbe u *Linuxu* upisujemo u naredbenu liniju iza prompta.

Kraj unosa naredbe označavamo pritiskom na tipku **[Return]**. Ta je tipka ujedno znak za kraj retka i prelazak u novi.

Osim u posebnim slučajevima (koji su objašnjeni u poglavlju 6. *Ljuska Bash*), kraj retka ujedno predstavlja i kraj naredbe.

Primjer vrlo jednostavne naredbe je **date**. Ta naredba ispisuje datum i vrijeme:

```
$ date
Mon Sep 16 10:52:32 CEST 2013
$
```

Uočite da smo naredbu `date` upisali iza prompta (koji je u ovom slučaju vrlo jednostavan, samo znak `$`). Sustav je u novi red ispisao datum i vrijeme. Nakon toga je ponovno ispisao prompt kao znak da je završio s izvođenjem naše naredbe i da je spreman prihvatiti novu naredbu.

Linux razlikuje mala od velikih slova. Tako su `date` i `DATE` (ili `DaTe`) različite naredbe.

U pravilu, sve se naredbe u *Linuxu* pišu malim slovima.

Neke naredbe (većina njih) omogućavaju nam da upisivanjem dodatnih parametara mijenjamo način na koji se one izvode.

Parametri mogu biti **opcije i argumenti**.

Pojedine dijelove naredbi (ime, opcije i argumente) međusobno odvajamo pritiskom na razmaknicu.

Na primjer, upoznali smo naredbu `date` koja standardno ispisuje datum i vrijeme. Ako želimo izmijeniti datum i vrijeme na sustavu, također se možemo poslužiti naredbom `date`, ali tada moramo promijeniti način na koji se ona izvodi. To radimo tako da nakon imena naredbe dodamo dodatne parametre:

```
$ date -s "3 OCT 2013 18:00:00"
Thu Oct 3 CEST 18:00:00
$
```

Opcijom `-s` smo promijenili ponašanje naredbe tako da umjesto ispisa ona mijenja datum i vrijeme na sustavu, a argument `"3 OCT 2013 18:00:00"` predstavlja izmjenu koju želimo napraviti.

2.4. Promjena lozinke

Lozinka se mijenja naredbom **passwd**.

Prilikom izmjene prvo treba upisati staru lozinku, nakon toga treba upisati novu lozinku i potvrditi je ponovnim upisom.

Napomena

Linux razlikuje mala od velikih slova. U pravilu, sve se naredbe u *Linuxu* pišu malim slovima.

Napomena

Ime naredbe određuje što treba napraviti, opcijama se određuje kako to treba napraviti, a argumentima se obično upućuje na objekt koji je predmet obrade.

Primjer:

```
$ passwd
Changing password for tecaj01
(current) UNIX password:
New UNIX password:
Retype new UNIX password:
$
```

Administratori koji održavaju sustav mogu postaviti različita ograničenja na odabir lozinke. Administratori mogu:

- odrediti najmanju duljinu lozinke
- odrediti trajanje valjanosti lozinke
- odrediti složenost lozinke
- spriječiti ponavljanje lozinke ili njihovu međusobnu sličnost
- spriječiti da lozinke sadrže osobne podatke korisnika poput korisničkog imena ili osobnog imena.

U slučaju da se prilikom odabira nove lozinke ne poštuje neko od pravila koje je postavio administrator, sustav neće prihvatiti lozinku i tražit će upisivanje neke druge nove lozinke.

Napomena

Lozinke se tijekom upisa ne prikazuju na ekranu.

Napomena

Osnovne su smjernice za odabir lozinke:

- lozinke ne smiju sadržavati osobne podatke korisnika
- lozinke trebaju biti dugačke najmanje šest znakova
- lozinke trebaju sadržavati različite znakove – brojke i slova
- najbolje je da je odabir i redoslijed znakova u lozinci slučajan.

2.5. Odjavljivanje sa sustava

Naredba za odjavljivanje sa sustava je **logout**.

U određenim slučajevima moguće je odjaviti se i upisivanjem naredbe **exit** ili kombinacijom (istovremenim pritiskom) tipki **[Ctrl]** i **[D]** (ta se kombinacija obično piše kao **[Ctrl-D]**).

Napomena

Na kraju rada obavezno se treba odjaviti.

2.6. Naredbe **echo** i **clear**

Naredba **echo** na ekran ispisuje zadane argumente.

Naredba **clear** briše sadržaj ekrana.

Sintaksa:

```
echo [STRING ...]
clear
```

Primjer:

```
$ echo Pozdrav svima
Pozdrav svima
$
```

2.7. Informacije o korisnicima na sustavu: **who**, **w**, **finger** i **last**

Naredba **who** ispisuje popis korisnika koji trenutačno rade na sustavu te podatke o tome od kada rade na sustavu i od kuda su se prijavili na sustav.

who am i ispisuje podatke samo o korisniku koji je pokrenuo naredbu.

Naredba **w** također ispisuje popis korisnika koji trenutačno rade na sustavu, ali s nekim dodatnim podacima (na primjer, kojim se programom korisnik trenutačno koristi i prije koliko je vremena upisao nešto na terminalu). Na vrhu ispisa bit će navedeni neki statistički podaci vezani uz sustav.

Naredbi **w** kao argument može se zadati ime nekog korisnika. Tada će se ispisati podaci samo za tog korisnika.

Sintaksa:

```
who [am i]
w [KORISNIK]
```

Primjeri:

```
$ who
root      pts/1    Sep 12 08:15  (pc1.srce.hr)
tecaj01   pts/3    Sep 12 10:23  (pc4.srce.hr)
$ who am i
tecaj01   pts/3    Sep 12 10:23  (pc4.srce.hr)
$
```

Naredba **finger**, ako je zadamo bez dodatnih argumenata, također ispisuje popis korisnika koji trenutačno rade na sustavu.

Ako kao argument naredbe **finger** navedemo neko korisničko ime, sustav će ispisati podatke o tom korisniku, neovisno je li taj korisnik trenutačno prijavljen na sustav ili nije.

Sintaksa:

```
finger [KORISNIK]
```

Primjeri:

```
$ finger
$ finger tecaj01
```

Napomena

Zapisi o prijavama na sustav (*log*) povremeno se brišu.

Stoga se na kraju popisa koji ispisuje naredba **last** nalazi naveden podatak (datum i vrijeme) od kada su na sustavu zabilježeni podaci o prijavama.

Naredba **last** ispisuje popis svih prijavljivanja na sustav.

Ukoliko se kao argument naredbe **last** navede neko korisničko ime, ispisat će se podaci samo za tog navedenog korisnika.

Sintaksa:

```
last [KORISNIK]
```

Primjeri:

```
$ last
$ last tecaj01
```


2.8. Komunikacija s drugim korisnicima: `write` i `mesg`

Naredba `write` ispisuje poruku na ekran korisnika čije je ime zadano kao argument. Poruku upisujemo nakon unosa naredbe, redak po redak. Kraj poruke označava se unosom kontrolnog znaka `[Ctrl-D]`.

Naredbom `mesg` određujemo smiju li se poruke drugih korisnika ispisivati na našem ekranu (kao argument treba upisati `y` ili `n`). Ukoliko naredbu napišemo bez argumenta, ispisat će nam se trenutna postavka.

Sintaksa:

```
write KORISNIK [TTY]
mesg [y|n]
```

Primjeri:

```
$ write tecaj01
Pozdrav. Kako si?
[Ctrl-D]
$

$ who -T
...
marko      + pts/0          2013-12-12 09:17 (10.0.2.2)
marko      + pts/1          2013-12-12 09:28 (10.0.2.2)
...
$ write marko pts/1
...
[Ctrl-D]
$
```

Napomena

Drugi argument naredbe `write` označava (virtualni) terminal na kojem radi korisnik kojem želimo poslati poruku.

To je korisno ako korisnik kojem želimo poslati poruku radi na više terminala istovremeno, odnosno ako je prijavljen na sustav više puta. U tom slučaju navođenjem drugog argumenta možemo odabrati na koji terminal želimo poslati poruku korisniku.

Oznake terminala na kojima radi korisnik kojem želimo poslati poruku možemo dobiti naredbom `who`.

Ako naredbi `who` dodamo opciju `-T` pokazat će se i dozvoljavaju li korisnici s popisa ispis poruka drugih korisnika na svojim ekranima.

2.9. Sustav pomoći: `man` i `info`

Linux ima dva sustava pomoći. Prvi sustav čine takozvane **man-stranice** (*man* je kratica od engleske riječi *manual*). Riječ je o dokumentaciji koja opisuje naredbe i druge dijelove sustava, a koja je podijeljena na poglavlja (sekcije) i stranice.

Navedena je dokumentacija na sustavu pohranjena u elektroničkom obliku, a dostupna je pomoću naredbe `man`.

Najjednostavniji način korištenja naredbe `man` je da se upiše njeno ime, a kao argument navede ime neke naredbe. U tom će nam slučaju sustav ispisati pripadajuću stranicu koja opisuje tu naredbu.

Neke naredbe ili općenito neke ključne riječi mogu biti opisane u više od jednog poglavlja (sekcije). Standardno će naredba `man` ispisati samo stranicu koju je prvo pronašla pretražujući dokumentaciju na sustavu.

Ako želimo da nam se ispišu sve stranice (iz različitih poglavlja), trebamo navesti opciju `-a`.

Ako želimo da nam se ispiše stanica iz nekog konkretnog poglavlja, tada kao argument trebamo navesti broj poglavlja.

Ako ne znamo ime naredbe za koju tražimo dodatne informacije, možemo pretraživati sustav pomoći pomoću ključne riječi. U tom slučaju kao opciju trebamo navesti **-k**, a kao argument željenu ključnu riječ. Rezultat će biti popis svih stranica koje u sebi sadrže navedenu ključnu riječ.

Sintaksa:

```
man [-a | POGLAVLJE] NAREDBA
man -k KLJUČNA_RIJEČ
```

Primjeri:

```
$ man shutdown
$ man -a shutdown
$ man -k shutdown
```

Prilikom čitanja man-stranica možemo koristiti sljedeće tipke:

- kursorke tipke - za kretanje kroz ispisane stranice
- razmaknicu (**[Space]**) - za prelazak na sljedeću stranicu
- **[p]** - za povratak na prethodnu stranicu
- **[q]** - za kraj ispisa.

Naredba **info** omogućava pristup do dokumenata koji čine drugi sustav pomoći, takozvane **info-stranice**. Ovaj se sustav pomoći sastoji od hipertekstualnih dokumenata. Dokumentaciju u ovom formatu imaju samo neki, većinom noviji programi posebno pisani upravo za *Linux*. Ukoliko ne postoji dokumentacija za traženi pojam među info-stranicama, naredba **info** će ispisati odgovarajuću man-stranicu (ako ona postoji).

Sintaksa:

```
info NAREDBA
```

Primjer:

```
$ info info
```

2.10. Vježba: Unos naredbi, komunikacija s drugim korisnicima i završetak rada na sustavu

1. Prijavite se na sustav, ukoliko već niste prijavljeni.
2. Provjerite izgled prompta.
3. Upišite naredbu **ls**.

Proučite rezultat. Po čemu znate da je naredba **ls** završila i da možete upisati novu naredbu?

4. Upišite sljedeće naredbe:

```
ls -l
ls /
ls -l /
```

Proučite rezultat. Kako su opcije i argumenti promijenili način rada naredbe `ls`?

5. Upišite `ls`.

Što se dogodilo?

6. Odaberite i postavite novu lozinku.

Odaberite novu lozinku u skladu s preporukama i dobro je zapamtite. Nakon toga promijenite lozinku.

7. Odjavite se sa sustava.

8. Pokušajte se prijaviti na sustav, prvo sa starom, a onda s novom lozinkom.

Prilikom prijave na sustav prvo upišite svoje korisničko ime i **staru lozinku**. Što se dogodilo? Nakon toga upišite **novu lozinku**. Je li sve prošlo u redu?

9. Provjerite jesu li na sustavu ispravno postavljeni datum i vrijeme.

10. Pomoću naredbe `who` prebrojite koliko korisnika trenutačno radi na sustavu.

Jesu li svi korisnici koji trenutačno rade na sustavu ujedno i polaznici tečaja?

11. Upoznajte se sa svojim susjedom na tečaju.

Koje je korisničko ime Vašeg susjeda? Je li on trenutačno prijavljen na sustav?

12. Provjerite je li dozvoljen ispis poruka drugih korisnika na Vašem ekranu. Ako nije, dozvolite ispis.

13. Zamolite susjeda da naredbom `write` pošalje kratku poruku na Vaš ekran. Pošaljite i Vi njemu poruku na ekran.

14. Dogovorite se sa svojim susjedom da istovremeno upišete naredbu `mesg n` i ponovite prethodni korak u vježbi.

Što se dogodilo?

15. Proučite koje podatke o sebi možete doznati pomoću naredbe `finger`.

Koji je Vaš početni direktorij? Koji je ime stroja s kojeg pristupate na sustav?

16. Pomoću naredbe `last` provjerite koji se korisnik posljednji prijavio na sustav.

Od kada sustav bilježi prijavljivanja?

17. Upišite naredbu `w` i proučite rezultat.

Je li sustav jako opterećen? Postoji li neki polaznik tečaja koji ne rješava ovu vježbu?

18. Provjerite imaju li pojmovi `passwd`, `gzip` i `coreutils` svoje man- i info-stranice.

Koji od navedenih pojmova imaju svoju man-stranicu, a koji imaju svoju

info-stranicu? Po čemu možete prepoznati na naredba `info` ispisuje man-stranicu, a ne info-stranicu?

19. Na kraju, obrišite sadržaj ekrana i pomoću naredbe `echo` ispišite „Kraj vježbe!“.

2.11. Pitanja za ponavljanje

1. Što je prompt? Koji je uobičajeni izgled prompta?
2. Koji su standardni elementi naredbi? Što određuje ime naredbe, a što određuju opcije i argumenti?
3. Kojom se naredbom mijenja lozinka?
4. Koje uvjete treba zadovoljavati dobro odabrana lozinka?
5. Kojom se naredbom odjavljuje sa sustava? Zašto se na kraju rada obavezno treba odjaviti?
6. Koja je razlika između naredbi `echo` i `write`?
7. Čemu služi naredba `mesg`?
8. Kojom naredbom brišemo sadržaj svog ekrana?
9. Koja je razlika između naredbi `who` i `finger`?
10. Koje podatke o korisnicima možemo dobiti pomoću naredbe `finger`?
11. Čemu služe naredbe `man` i `info`?
12. Kojom opcijom naredbe `man` pretražujemo sustav pomoći?

3. Sustav datoteka

3.1. Općenito o sustavu datoteka

Sustav datoteka je dio operacijskog sustava koji omogućava pohranjivanje i upravljanje datotekama i direktorijima.

Datoteka je logička cjelina koja služi za pohranjivanje podataka unutar sustava datoteka.

Direktorij (mapa) je posebna vrsta datoteke koja sadrži datoteke i druge direktorije (poddirektorije).

Linux ima **hijerarhijski sustav datoteka** (direktoriji mogu imati poddirektorije).

Vršni direktorij se još zove i *root*-direktorij i označava se znakom `/`.

Na *Linuxu* može postojati samo jedan sustav datoteka i samo jedan vršni direktorij. Ako na sustavu postoji više diskova ili diskovnih particija, oni su na *Linuxu* uključeni u jedan te isti sustav datoteka.

Na *Linuxu* su svi hardverski uređaji (pisači, mrežne kartice, uređaji za trajnu pohranu podataka, USB-priključci i drugo) također prikazani kao datoteke i može im se pristupati na isti način kao i običnim datotekama na disku. Operacijski sustav se brine o tome da se podaci prenesu navedenim uređajima na odgovarajući način.

Dakle, datoteke na *Linuxu* mogu biti:

- obične
- direktoriji
- specijalne (uređaji, cjevovodi za komunikaciju među procesima i drugo).

Sustav datoteka na *Linuxu* sastoji se od tisuća datoteka i direktorija. Uobičajena je organizacija sljedeća:

```
/bin    – programi koje koriste i administratori i korisnici
/dev    – datoteke koje predstavljaju hardverske uređaje
/etc    – konfiguracijske datoteke
/home   – osobni direktoriji korisnika
/sbin   – sistemski programi
/tmp    – privremene datoteke
/usr    – korisnički programi, dokumentacija i biblioteke
/var    – sistemski zapisi i druge promjenjive datoteke.
```

Napomena

Na različitim sustavima *Linux* (koji se popularno zovu distribucije) organizacija sustava datoteka može biti različita.

3.2. Imena datoteka

Datoteke na *Linuxu* mogu imati imena duljine do 255 znakova. Mogu se koristiti svi mogući znakovi, premda je preporučljivo da se koriste samo alfanumerički znakovi.

Ne preporuča se uporaba hrvatskih dijakritičkih znakova te metaznakova koji imaju posebno značenje za sustav:

* ? > < / ; ! [] | \ ' " () { }

Primjeri dobro odabranih imena datoteka:

```
moja_datoteka
izvjestaj.txt
program3.c
primjer-123a
Poruka-za-Marka
.sifra
```

Primjeri loše odabranih imena datoteka:

```
moja datoteka
izvještaj.txt
+program.c
primjer*123a
Račun za restoran "Tri školjke"
izvještaj/poglavlje2
```

Svaki direktorij sadrži i dvije „posebne“ datoteke. To su `.` (točka) i `..` (dvije točke).

- `.` – oznaka za radni direktorij
- `..` – oznaka za nadređeni direktorij.

Imena datoteka koja počinju znakom `.` (na primjer: `.profile`) sustav smatra **skrivenim**. To znači da kad se, na primjer, ispisuje sadržaj direktorija pomoću naredbe `ls` i ne navedu se dodatne opcije (o čemu će biti riječi u nastavku ovog poglavlja) imena datoteka koja počinju znakom `.` se ne pojavljuju u popisu.

3.3. Ispis radnog direktorija: `pwd`

Naredba `pwd` ispisuje koji je naš radni direktorij.

Sintaksa:

```
pwd
```

Primjer:

```
$ pwd
/home/tecaj01
$
```

3.4. Apsolutne i relativne staze

Staza (*pathname*) je niz imena odijeljenih znakom / (kosa crta, *slash*) koja opisuju put ili rutu koju je potrebno proći kroz sustav datoteka da bi se došlo do željene datoteke ili direktorija.

Posljednje ime u stazi može biti ime datoteke ili direktorija. Sva ostala imena u stazi moraju biti nazivi direktorija.

Svaka staza koja ne počinje znakom / je **relativna** i opisuje put od radnog direktorija do krajnjeg cilja.

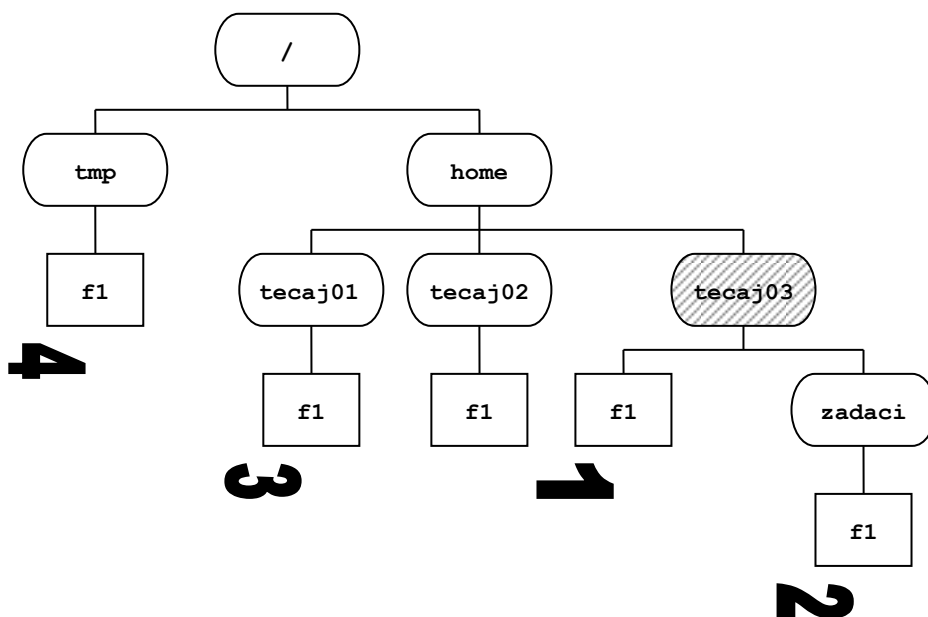
Staze koje počinju znakom / su **apsolutne** i opisuju rutu od vršnog direktorija do krajnjeg cilja.

Ako se koristi na početku staze, znak ~ (tilda) označava početni direktorij.

Potpuno ime neke datoteke ili direktorija je apsolutna staza do te datoteke koja završava s (kratkim) imenom te datoteke ili direktorija. Na sustavu ne može postojati više datoteka (ili direktorija) s istim potpunim imenom.

Primjeri apsolutnih i relativnih staza:

Pretpostavimo da sljedeća slika opisuje naš sustav datoteka i da je naš radni direktorij /home/tecaj03 (označen sjenčanjem):



U sljedećoj se tablici nalaze relativne i apsolutne staze za četiri označena slučaja:

Relativna staza	Apsolutna staza
1. f1	1. /home/tecaj03/f1
2. zadaci/f1	2. /home/tecaj03/zadaci/f1
3. ../tecaj01/f1	3. /home/tecaj01/f1
4. ../../tmp/f1	4. /tmp/f1

3.5. Promjena radnog direktorija: `cd`

Naredba `cd` omogućava promjenu radnog direktorija.

Dovoljno je kao argument navesti ime direktorija za koji želimo da bude novi radni direktorij.

Vrijede sljedeći **posebni slučajevi**:

- ako se kao argument navede znak `-` (minus), tada će se za novi radni direktorij postaviti direktorij koji je bio radni direktorij neposredno prije trenutnog
- ako se ne navede nikakav argument, tada će se za novi radni direktorij postaviti početni direktorij.

Ime novog radnog direktorija može se zadati bilo relativnom, bilo apsolutnom stazom.

Sintaksa:

```
cd [DESTINACIJA]
```

Primjeri:

```
$ cd /tmp
$ cd /usr
$ pwd
/usr
$ cd -
$ pwd
/tmp
$ cd
$ pwd
/home/tecaj01
$
```

3.6. Ispis sadržaja direktorija: `ls`

Naredba `ls` ispisuje sadržaj direktorija. Dodatni podaci koji se pri tome ispisuju određuju se opcijama.

Sintaksa:

```
ls [opcije] [DIREKTORIJ_ILI_DATOTEKA]
```

Ako se umjesto imena direktorija navede ime neke datoteke, ispisat će se podaci o toj datoteci.

Ako se ne navede nikakav argument, podrazumijeva se da se radi o radnom direktoriju. Moguće je zadati i više argumenata (popis direktorija čiji sadržaj želimo ispisati).

Najčešće korištene **opcije** naredbe `ls` su sljedeće:

- `-l` - ispis dodatnih podataka o datotekama i poddirektorijima
- `-a` - ispis svih datoteka i direktorija, uključivši i one skrivene (čija imena počinu točkom)

- d - umjesto sadržaja ispisuje svojstva direktorija (najčešće se koristi zajedno s opcijom -l)
- F - na kraj imena direktorija dodaje znak /, a na kraj imena datoteke koja se može izvoditi dodaje znak *
- R - rekurzivno ispisuje sadržaj zadanog direktorija i svih poddirektorija.

Više opcija moguće je navesti zajedno (na primjer: `ls -al`).

Primjeri:

```
$ ls
doc test
$ ls -a
.profile doc test
$ ls abc
ls: cannot access abc: No such file or directory
$ ls /
bin  etc  lib  tmp  var
dev  home sbin usr
$
```

Primjer ispisa naredbe `ls`:

```
$ ls -la
drwxr-x---  5 tecaj01 tecaj 4096 Sep 29 16:22 .
drwxr-xr-x 26 root    root 4096 Aug 10 09:27 ..
-rw-r--r--  1 tecaj01 tecaj 191  Feb  7  2005 .bash_profile
drwx-----  3 tecaj01 tecaj 4096 Sep 29 16:22 privatno
-rw-r--r--  1 tecaj01 tecaj 2332 Oct 21  2005 zadatak.txt
$
```

A **B** **C**

- A** - veličina u bajtovima
- B** - datum/vrijeme zadnje modifikacije
- C** - ime

Ispis naredbe `ls` bit će detaljnije objašnjen u sljedećem poglavlju.

3.7. Stvaranje i uklanjanje direktorija: `mkdir` i `rmdir`

Naredba `mkdir` stvara, a `rmdir` uklanja direktorije sa sustava.

Da bi se direktorij mogao obrisati on mora biti prazan (ne smije sadržavati neki poddirektorij ili datoteku).

Radni direktorij (direktorij u kojem se trenutno nalazimo i iz kojeg pozivamo naredbu `rmdir`) ne može se obrisati.

Obje naredbe mogu se zadati s opcijom `-p`. U tom slučaju stvorit će se ili ukloniti svi direktoriji navedeni kao dio puta koji je naveden kao argument.

Na primjer, naredba `mkdir -p test/abc` prvo će, ako on ne postoji, stvoriti direktorij `test`, a nakon toga u njemu poddirektorij `abc`.

Sintaksa:

```
mkdir [-p] DIREKTORIJ
rmdir [-p] DIREKTORIJ
```

Primjeri:

```
$ mkdir doc
$ mkdir test/abc
mkdir: cannot create directory 'test/abc': No such
file or directory
$ mkdir -p test/abc
$ rmdir test
rmdir: test: Directory not empty
$ rmdir doc
$
```

3.8. Stvaranje datoteka: touch

Naredba **touch** stvara datoteku sa zadanim imenom. Tako nastale datoteke su prazne, veličine 0 bajtova.

Ako datoteka čije je ime zadano kao argument već postoji, promijenit će se samo datum njene zadnje modifikacije. Sadržaj datoteke neće biti izmijenjen.

Sintaksa:

```
touch DATOTEKA
```

Primjer:

```
$ ls test.txt
ls: test.txt: No such file or directory
$ touch test.txt
$ ls test.txt
test.txt
$
```

Napomena

Obrisane datoteke (i direktorije) više nije moguće vratiti.

3.9. Brisanje datoteka: rm

Naredba **rm** uklanja datoteke sa sustava.

Sintaksa:

```
rm [-i] DATOTEKA
rm -r[i] DIREKTORIJ
```

Opcijom **-r** zadaje se rekurzivno uklanjanje datoteka i direktorija. Na ovaj način možemo ukloniti i direktorije koji nisu prazni.

Ukoliko se zada opcija **-i**, naredba **rm** će prije uklanjanja bilo koje datoteke zatražiti potvrdu. Na nekim je sustavima ova opcija postavljena kao podrazumijevana (nije je potrebno posebno navoditi). U tom se slučaju njezin učinak poništava navođenjem opcije **-f**.

Primjeri:

```
$ rm test
rm: cannot remove 'tmp': Is a directory
$ rmdir test
rmdir: test: Directory not empty
$ rm -r test
$
```

3.10. Kopiranje i premještanje datoteka i direktorija: **cp i mv**

Naredba **cp** služi za kopiranje datoteka i direktorija.

Pri tome vrijedi sljedeće:

- treba navesti barem dva argumenta: izvor i destinacija
- ako se kopira samo jedna datoteka tada destinacija može biti ime nove datoteke ili ime direktorija u koji se kopira izvorna datoteka
- ako je kao cilj zadan direktorij, izvorna će se datoteka kopirati u zadani direktorij i sačuvati originalno ime
- kada se odjednom kopira više datoteka, kao cilj se mora zadati direktorij
- datoteka ne može biti kopirana u samu sebe
- direktoriji se kopiraju navođenjem opcije **-r**.

Sintaksa:

```
cp DATOTEKA DESTINACIJA
cp DATOTEKA [DATOTEKA ...] DIREKTORIJ
cp -r DIREKTORIJ DESTINACIJA
```

Naredba **mv** služi za premještanje ili preimenovanje datoteka i direktorija.

Vrijedi sljedeće:

- treba navesti barem dva argumenta: izvor i destinaciju premještanja ili preimenovanja
- ako je prvi argument ime datoteke tada destinacija može biti novo ime datoteke ili ime direktorija u koji se premješta izvorna datoteka:
 - ako je kao destinacija zadan direktorij, izvorna će se datoteka premjestiti u zadani direktorij i sačuvati originalno ime
 - ako je kao destinacija zadano novo ime datoteke, tada će se originalna datoteka preimenovati ako je cilj u istom direktoriju ili premjestiti i dobiti novo ime ako je cilj u drugom direktoriju
- kada se odjednom premješta više datoteka, kao destinacija se mora zadati direktorij
- za premještanje i preimenovanje direktorija vrijede ista pravila kao i za datoteke.

Sintaksa:

```
mv DATOTEKA DESTINACIJA
mv DATOTEKA [DATOTEKA ...] DIREKTORIJ
mv DIREKTORIJ DESTINACIJA
```

3.11. Ispis sadržaja datoteka: `cat`, `more` i `less`

Naredbe `cat`, `more` i `less` ispisuju sadržaj datoteka.

Sintaksa:

```
cat DATOTEKA [DATOTEKA...]
more DATOTEKA [DATOTEKA...]
less DATOTEKA [DATOTEKA...]
```

Naredba `cat` je najjednostavnija. Ona redom ispisuje sadržaj datoteke ili datoteka koje su navedene kao argumenti.

Naredbe `more` i `less` ispisuju sadržaj datoteka koje su navedene kao argumenti ekran po ekran. Prilikom pregledavanja sadržaja datoteka moguće je koristiti sljedeće komande (tipke na tipkovnici):

[q]	- završetak rada
[Space]	- prikaz sljedeće stranice
[Return]	- prikaz sljedećeg reda
[b]	- prethodna stranica
[/]	- pretraživanje.

3.12. Zamjenski znakovi

Najvažniji zamjenski znakovi su:

*	- zamjenjuje bilo koji broj znakova
?	- zamjenjuje točno jedan znak
[]	- označava zamjenu za točno jedan znak koji odgovara znakovnom nizu zadanom unutar zagrada

Nizovi se zadaju nabranjem ili kao intervali.

Unutar zagrada moguće je koristiti i znak negacije `!` (uskličnik).

Zamjenski se znakovi mogu koristiti prilikom zadavanja argumenata u naredbama. Na primjer, naredba `ls a*` će ispisati imena svih datoteka u radnom direktoriju čija imena započinju slovom `a`.

Primjer:

Pretpostavimo da se u našem radnom direktoriju nalazi sljedećih devet datoteka:

`a`, `a1`, `a2`, `a3`, `ab`, `abc`, `b1`, `b2`, `c1` i `d3`.

Sljedeća tablica pokazuje što bi ispisala naredba `ls` u kombinaciji s nekoliko različitih argumenata zadanih pomoću zamjenskih znakova:

naredba	ispisuje	ne ispisuje
<code>ls a*</code>	a a1 a2 a3 ab abc	b1 b2 c1 d3
<code>ls a?</code>	a1 a2 a3 ab	a abc b1 b2 c1 d3
<code>ls [ad]*</code>	a a1 a2 a3 ab abc d3	b1 b2 c1
<code>ls [a-c]*</code>	a a1 a2 a3 ab abc b1 b2 c1	d3
<code>ls [!a]*</code>	b1 b2 c1 d3	a a1 a2 a3 ab abc

3.13. Pretraživanje sustava datoteka: `find`

Sustav datoteka se pretražuje naredbom `find`.

Sintaksa:

```
find [STAZA...] [UVJET]
```

`STAZA` je direktorij od kojeg počinje pretraživanje, a `UVJET` je izraz kojim se određuje koje se datoteke traže. Najčešće je to `-name` (pretraživanje po imenu datoteke).

Primjeri:

```
$ find / -name abc
$ find . -name program.c
```

3.14. (π) Vježba: Rad s datotekama i direktorijima

1. U Vašem početnom direktoriju nalazi se poddirektorij `poglavlje3`. Neka taj direktorij postane Vaš radni direktorij.

Koja je apsolutna staza do tog direktorija? Koja je relativna staza do tog direktorija od Vašeg početnog direktorija?

Kako ćete provjeriti nalazite li se u pravom direktoriju?

2. Koje se datoteke nalaze u tom direktoriju?

Koje su veličine navedene datoteke? Ima li navedeni direktorij poddirektorije?

3. Pomoću naredbe `ls`:

- a. ispišite imena datoteka koja završavaju s nastavkom `.c`
- b. ispišite imena datoteka koja u svom imenu sadrže slovo `m`
- c. ispišite imena datoteka koja ne započinju slovom `p`
- d. ispišite imena datoteka koja započinju slovima `x` ili `v`.

4. Zabilježite vremena zadnje modifikacije datoteka `zadatak.txt` i `rezultati.dat`.

5. Kreirajte poddirektorij `abc`. U njega kopirajte datoteku `zadatak.txt` i premjestite datoteku `rezultati.dat`.

Ispišite sadržaj direktorija `abc`. Koja su vremena modifikacije datoteka u tom direktoriju? Zabilježite navedene podatke. Razlikuju li se ti podaci od onih koje ste zabilježili u prethodnom koraku?

6. Neka poddirektorij `abc` bude Vaš radni direktorij. Ispišite sadržaj datoteke `rezultati.dat` pomoću naredbe `cat`. Pročitajte sadržaj datoteke `zadatak.txt` pomoću naredbe `less`.

7. Pokušajte naredbom `touch` stvoriti datoteku `rezultati.dat`.

Je li datoteka postojala od prije? Proučite njenu veličinu i vrijeme zadnje modifikacije. Što se promijenilo? Je li se promijenio njezin sadržaj?

8. Potražite datoteku `zadatak.txt` ispod svog početnog direktorija.

3.15. Pitanja za ponavljanje

1. Koje dužine mogu biti imena datoteka na *Linuxu*?
2. Koje znakove nije preporučljivo koristiti u imenima datoteka?
3. Što su apsolutni i relativni put?
4. Što su skrivene datoteke?
5. Što predstavljaju posebna imena datoteka: `.` i `..`?
6. Kako najlakše postavljamo početni direktorij za radni direktorij?
7. Kojim naredbama stvaramo i uklanjamo direktorije?
8. Kako možemo dobiti popis svih datoteka u direktoriju?
9. Koji su zamjenski znakovi i kako ih koristimo?
10. Koja je razlika između naredbi `cp` i `mv`?
11. Kako možemo ukloniti direktorije koji nisu prazni?
12. Koja je razlika između naredbi `cat` i `less`?

4. Prava pristupa datotekama i direktorijima

4.1. Naredba `id`

Svakom korisniku dodijeljen je jedinstven **korisnički broj** (*user ID*, **uid**).

Također, svaki je korisnik pridružen jednoj ili više **skupina korisnika**. Skupine također imaju svoje jedinstvene brojeve (*group ID*, **gid**).

Skupine služe za lakše dodjeljivanje istih korisničkih prava većem broju korisnika.

Svaki je korisnik pridružen barem jednoj skupini i ta je skupina za tog korisnika osnovna skupina kojoj pripada.

Korisnički broj je osnovni način na koji sustav razlikuje korisnike. Korisnički broj, osnovnu skupinu i pripadnost drugim skupinama određuje administrator sustava.

Naredba `id` (bez dodatnih argumenata) ispisuje pripadajući korisnički broj, broj osnovne skupine i popis svih skupina kojima pripada korisnik koji je pokrenuo naredbu.

Ako se kao argument zada ime nekog korisnika, tada se ispisuju podaci o tom korisniku.

Sintaksa:

```
id [KORISNIK]
```

Primjeri:

```
$ id
uid=500(tecaj01) gid=900(tecaj) groups=900(tecaj)
$ id tecaj02
uid=501(tecaj02) gid=900(tecaj) groups=900(tecaj)
```

4.2. Razine prava pristupa: *user*, *group* i *other*

Svaka datoteka (ili direktorij) ima svog vlasnika. Vlasnik je obično korisnik koji je stvorio tu datoteku.

Vlasnik je također korisnik koji određuje prava pristupa za navedeni objekt.

Svaka datoteka ili direktorij pripadaju i nekoj skupini korisnika. Najčešće je to ista skupina kojoj pripada i vlasnik, ali ne i nužno. Vlasnik određuje kojoj će skupini pripadati njegove datoteke i direktoriji.

Linux omogućava **tri razine dodjeljivanja prava pristupa** (*ugo*):

- u** - pravo pristupa koje se odnosi na vlasnika (*user*)
- g** - pravo pristupa koje se odnosi na skupinu (*group*)
- o** - pravo pristupa koje se odnosi na ostale (*other*)

Za svaku od navedenih razina moguće je zasebno određivati prava pristupa.

4.3. Vrste prava pristupa: *read*, *write* i *execute*

Prava pristupa na datoteke i direktorije koja se mogu dodijeliti korisnicima su:

- r** - pravo čitanja (*read*)
- w** - pravo izmjene sadržaja (*write*)
- x** - pravo izvršavanja (*execute*).

Konkretni smisao navedenih prava pristupa ovisi o vrsti objekta na koji se odnosi (tj. je li riječ o datoteci ili direktoriju):

	read	write	execute
datoteke	Sadržaj se može čitati (<i>more</i> , <i>cat</i>).	Sadržaj se može mijenjati (u datoteku se može pisati).	Datoteka se može koristiti kao naredba.
direktoriji	Sadržaj se može čitati (<i>ls</i>).	Sadržaj se može mijenjati (<i>rm</i> , <i>cp</i> , <i>mv</i>).	Može postati radni direktorij (<i>cd</i>).

Prava pristupa možemo pročitati iz ispisa koji daje naredba `ls -l`.

Primjer:

```
$ ls -la
```

drwxr-x---	5	tecaj01	tecaj	4096	Sep 29 16:22	.
drwxr-xr-x	26	root	root	4096	Aug 10 09:27	..
-rw-r--r--	1	tecaj01	tecaj	191	Feb 7 2005	.bash_profile
drwx-----	3	tecaj01	tecaj	4096	Sep 29 16:22	privatno
-rw-r--r--	1	tecaj01	tecaj	2332	Oct 21 2005	zadatak.txt

1 2 3 4 5 6 7 8

- 1** - vrsta objekta (znak – označava obične datoteke, a slovo *d* označava direktorije)
- 2** - prava pristupa za vlasnika, skupinu i ostale korisnike - tri skupine prava pristupa sa po tri oznake (**rwX**)
- 3** - broj poveznica (sistemski podatak)
- 4** - ime vlasnika
- 5** - ime skupine
- 6** - veličina u bajtovima
- 7** - datum i vrijeme zadnje modifikacije
- 8** - ime datoteke ili direktorija

Opis prava pristupa za vlasnika, skupinu i ostale korisnike je oblika:

rwX rwX rwX

Prva tri stupca odnose se na prava pristupa vlasnika, sljedeća tri na skupinu, a posljednja tri na ostale korisnike. Svaki od stupaca odgovara jednom od prava pristupa (*read*, *write* ili *execute*). Ako je određeno pravo pristupa dodijeljeno (omogućeno), tada se na odgovarajućem mjestu nalazi slovo koje odgovara toj vrsti pristupa (*r*, *w* ili *x*). U suprotnom se nalazi znak – (crta).

Primjeri prava pristupa:

`rw-rw-rwx` - svi imaju pravo čitanja, pisanja i izvršavanja
`rw-r--x--x` - vlasnik ima sva prava, a skupina i svi ostali imaju samo prava čitanja i izvršavanja
`rw-r-----` - vlasnik ima pravo čitanja i pisanja, skupina ima samo pravo čitanja, a ostali korisnici nemaju nikakvo pravo pristupa.

4.4. Postavljanje prava pristupa pomoću simboličkih oznaka

Prava pristupa postavljamo naredbom `chmod`. Ova naredba omogućava postavljanje prava pristupa na dva načina. Prvi je način pomoću **simboličkih oznaka** za razine i vrste prava pristupa, a koje smo već imali prilike upoznati:

- *u*, *g* i *o* za razine pristupa (te *a* koji označava sve razine)
- *r*, *w* i *x* za vrste prava pristupa.

Vrsta izmjene se zadaje operatorima: + (dodaj), – (oduzmi) i = (izjednači).

Sintaksa:

`chmod POSTAVKA [, POSTAVKA...] DATOTEKA`

POSTAVKA		
tko	vrsta promjene	vrsta pristupa
<i>u</i> (vlasnik)	+ (dodaj)	<i>r</i> (čitanje)
<i>g</i> (grupa)	– (oduzmi)	<i>w</i> (pisanje)
<i>o</i> (ostali)	= (izjednači)	<i>x</i> (izvršavanje)
<i>a</i> (svi)		

Različite postavke je moguće ulančavati, pri čemu se kao separator koristi zarez. Lijeva i desna strana postavke može sadržavati i više od jednog simbola.

Najjednostavniji oblik zadavanja promjene prava pristupa je niz od tri znaka, gdje prvi znak označava na koga se promjena odnosi, drugi je znak operator koji označava o kakvoj se promjeni radi, a treći je znak vrsta pristupa koja se mijenja (na primjer: *u+x* – vlasniku se dodaje pravo čitanja datoteke).

Primjeri:

```
$ chmod a=r abc
$ ls -l abc
-r--r--r-- 1 tecaj01 tecaj 2332 Oct 21 2013 abc
$ chmod u+wx abc
$ ls -l abc
-rwxr--r-- 1 tecaj01 tecaj 2332 Oct 21 2013 abc
$ chmod g+x,o-r abc
$ ls -l abc
-rwxr-x--- 1 tecaj01 tecaj 2332 Oct 21 2013 abc
$
```

prije	naredba	poslije
rw- r-- ---	chmod u-w,g-r	r-- --- ---
	chmod a+x	rwX r-X --X
	chmod ug+x	rwX r-X ---
	chmod a=rX	r-X r-X r-X

4.5. Postavljanje prava pristupa pomoću numeričkih oznaka

Drugi način postavljanja prava pristupa je preko numeričkih oznaka.

Svakoj vrsti pristupa dodijeljena je jedna numerička vrijednost:

```
read    = 4
write   = 2
execute = 1.
```

Pravo pristupa za svaku od tri razina pristupa (*user*, *group*, *other*) određuje se zbrajajući pripadajuće vrijednosti načina pristupa: ako je odgovarajuće pravo pristupa dozvoljeno, tada se ukupnom zbroju dodaje pripadajuća numerička vrijednost. Na taj način se svaka kombinacija prava pristupa može jednoznačno predstaviti brojem iz intervala od 0 do 7.

Konačan rezultat je niz od tri jednoznamenkasta broja kojim određujemo prava pristupa za sva tri nivoa.

Sintaksa:

```
chmod POSTAVKA DATOTEKA
```

Primjer:

Pretpostavimo da želimo postaviti prava pristupa na neku datoteku tako da vlasnik ima sva prava nad datotekom (može je čitati, pisati u nju i izvršavati), a da skupina i svi ostali korisnici datoteku mogu samo čitati i izvršavati.

Numeričku oznaku izračunat ćemo na sljedeći način:

Postavke za vlasnika datoteke: $rwX = 4+2+1 = 7$.

Postavke za skupinu: $r-X = 4 + 1 = 5$.

Postavke za ostale: $r-X = 4 + 1 = 5$.

Konačan rezultat je 755, a naredba glasi: `chmod 755 DATOTEKA`.

rwX r-X r-X		
vlasnik	skupina	ostali
rwX	r-X	r-X
↓	↓	↓
4+2+1	4+0+1	4+0+1
↓	↓	↓
7	5	5
<code>chmod 755 DATOTEKA</code>		

4.6. Naredba `umask`

Prilikom stvaranja datoteka, sustav obično za njih određuje maksimalna prava pristupa: `rw-rw-rw-`. U slučaju direktorija to je `rwXrwXrwX`. To znači da u tom slučaju svatko može mijenjati sadržaj novostvorenih datoteka, a u novostvorenim direktorijima svatko može stvarati i uklanjati datoteke.

Naredba `umask` omogućava izmjenu podrazumijevanih prava pristupa koje se dodjeljuju datotekama i direktorijima prilikom njihovog stvaranja. Podrazumijevana prava pristupa zadaju se **maskom** - nizom od tri znamenke.

Maska predstavlja vrijednosti koje će se prilikom stvaranja datoteke ili direktorija oduzeti od maksimalnih mogućih vrijednosti prava pristupa.

Ukoliko se naredba `umask` zada bez argumenta (bez nove vrijednosti maske) ispisat će se trenutna vrijednost maske.

Sintaksa:

`umask [MASKA]`

Primjer izračuna maske:

Pretpostavimo da želimo da kod stvaranja datoteke vlasnici imaju sva prava (dakle da im se ne oduzme niti jedno od maksimalnih prava), a da se članovima skupine kojoj pripada vlasnik oduzme pravo pisanja, a svim ostalim korisnici oduzmu i pravo pisanja i pravo čitanja datoteke. Tada ćemo masku izračunati na sljedeći način:

maksimalne postavke	rw- rw- rw-	6 6 6
željene postavke	rw- r-- ---	6 4 0
razlika		0 2 6
rezultat	<code>umask 026</code>	

Primjeri:

```

$ umask
000
$ touch nova1
$ ls -l nova1
-rw-rw-rw- 1 tecaj01 tecaj 0 Oct 21 12:22 nova1
    ↑
    nova datoteka ima prava pristupa: rw-rw-rw-

$ umask 022
$ umask
022
$ touch nova2
$ ls -l nova2
-rw-r--r-- 1 tecaj01 tecaj 0 Oct 23 12:23 nova2
    ↑
    nova datoteka ima drugačija prava pristupa: rw-r--r--

```

← početna vrijednost maske iznosi 000

← pomoću naredbe touch stvaramo datoteku nova1

← postavljamo novu vrijednost maske: 022

← na isti način kao i u prethodnom primjeru stvaramo novu datoteku nova2

Na mnogim je sustavima maska postavljena od strane administratora (putem konfiguracijskih datoteka) upravo na vrijednost 022.

4.7. Vježba: postavljanje prava pristupa

1. Provjerite koji su Vaš korisnički broj (UID) i broj skupine (GID).
2. U Vašem početnom direktoriju nalazi se poddirektorij `poglavlje4`. Neka taj direktorij postane Vaš radni direktorij.

Koje se datoteke nalaze u tom direktoriju? Koja su prava pristupa na te datoteke?

3. Pomoću naredbe `chmod` i služeći se simboličkim načinom zadavanja izmjena prava pristupa:

- dodajte svima pravo izvršavanja datoteke `primjer4-2`
- dodajte ostalim korisnicima pravo čitanja datoteke `primjer4-3`
- oduzmite skupini i ostalim korisnicima pravo izmjene sadržaja datoteke `primjer4-1`.

4. Pomoću naredbe `chmod` i koristeći numerički način zadavanja izmjena prava pristupa postavite pravo pristupa `rxr-x--` na radni direktorij.

5. Koja je vrijednost maske? Postavite vrijednost maske tako da je pravo pristupa na novostvorene datoteke `rw-rw----`.

Provjerite jeste li ispravno odabrali novu vrijednost maske tako da naredbom `touch` stvorite novu datoteku `nova.txt`. Provjerite ima li nova datoteka očekivana prava pristupa.

6. Ponovno postavite vrijednost maske na originalnu.

7. Naredba `find` omogućava traženje datoteka koje pripadaju određenom korisniku. Za to služi opcija `-user` (na primjer: `find / -user marko` ispisat će sve datoteke na sustavu koje pripadaju korisniku `marko`).

Potražite datoteke u direktoriju `/tmp` koje pripadaju korisniku `root`.

Ima li u tom direktoriju i neka koja pripada Vama?

4.8. Pitanja za ponavljanje

1. Čemu služi naredba `id`?
2. Koje su razine pristupa datotekama i direktorijima?
3. Koje su vrste pristupa datotekama i direktorijima?
4. Kako se simbolički zadaje promjena prava pristupa?
5. Kako se numerički postavlja pravo pristupa?
6. Čemu služi naredba `umask`?

5. Uređivač teksta *GNU nano*

5.1. Općenito o uređivačima teksta

Napomene

Uređivači teksta (editori) služe za pisanje i osnovno uređivanje teksta, poput pisanja programa, uređivanja konfiguracijskih datoteka i slično.

Uređivači teksta nisu isto što i programi za obradu teksta (poput programa *MS Word*).

Uređivači teksta (editori) su programi koji služe za pisanje i osnovno uređivanje teksta. Oni ne posjeduju napredne mogućnosti poput određivanja vrste fonta, veličine slova ili proreda između redaka, a koje imaju programi za obradu teksta.

Osnovne mogućnosti uređivača teksta su stvaranje, učitavanje i pohranjivanje tekstualnih datoteka, upisivanje i izmjena teksta te jednostavno pretraživanje teksta.

Na *Linuxu* možemo koristiti više uređivača teksta. Standardni uređivač teksta koji je sastavni dio svakog sustava je **Vi**. Riječ je o kompleksnom programu teškom za uporabu. Jednostavniji za uporabu je uređivač **GNU nano**. Ovaj je program u pravilu standardan dio svake distribucije *Linuxa*.

Osim ova dva uređivača teksta postoji i veliki broj drugih različitih uređivača teksta (jedan od popularnijih je program **Joe**). Neki od njih rade u tekstualnom okruženju poput programa *Vi* i *GNU nano*, a neki rade u grafičkom okruženju (na primjer **Kate** u okruženju KDE, a **Gedit** u okruženju *Gnome*).

5.2. Osnovna svojstva uređivača teksta *GNU nano*

Napomena

Program **GNU nano** standardno je dostupan na velikoj većini sustava zasnovanih na *Linuxu*. Ukoliko to nije slučaj sa sustavom na kojem radite, tada trebate zamoliti sistemskog administratora da instalira navedeni program.

Prije toga možete provjeriti postoji li na sustavu editor *Pico* – rad s tim editorom je gotovo identičan radu s editorom *GNU nano*.

Uređivač teksta *GNU nano* je jednostavan program. Da bi se moglo raditi s njime dovoljno je naučiti mali broj osnovnih naredbi koje se pokreću odgovarajućim kombinacijama tipki. Stoga je i neiskusnim korisnicima dovoljno samo kratko vrijeme uvježbavanja prije početka korištenja.

GNU nano je „klon“ jednostavnog editora **Pico** koji je bio dio programskog paketa za rad s elektroničkom poštom *Pine* i tako je stekao velik broj korisnika. Neki inačice sustava *Linux* umjesto editora *GNU nano* i dalje sadrže editor *Pico*. Sve funkcije editora *GNU nano* koje se spominju u ovom tečaju identične su funkcijama editora *Pico*.

Razlog za nastanak editora *GNU nano* je to što, premda je editor *Pico* u svojoj osnovi besplatan i slobodno dostupan softver, licenca kojom je određeno pravo uporabe programa *Pico* nije sukladna licenci GPL (*General Public License*).

Osim kopiranja gotovo svih funkcija programa *Pico*, program *GNU nano* sadrži i brojna proširenja.

5.3. Pokretanje programa

Program *GNU nano* se pokreće upisivanjem naredbe **nano**. Ukoliko se naredba upiše bez dodatnih argumenata, na ekranu će se pokazati prazna stranica u koju odmah možemo početi pisati tekst. Ukoliko se kao argument navede ime neke datoteke, *GNU nano* će učitati sadržaj te datoteke.

Sintaksa:

```
nano [DATOTEKA]
```

Primjeri:

```
$ nano
$ nano .profile
```

Unutar teksta obično se možemo kretati pomoću kursorских tipki koje se nalaze na tipkovnici. Ipak, može se dogoditi da sustav ne prepozna kursorске tipke. U tom slučaju možemo koristiti sljedeće kombinacije tipki:

```
[Ctrl-B] - lijevo
[Ctrl-F] - desno
[Ctrl-P] - gore
[Ctrl-N] - dolje
```

Ako želimo obrisati dio teksta, to možemo učiniti pomoću za to uobičajenih tipki na tipkovnici: **[Delete]** i **[Backspace]**. Ukoliko sustav na kojem radimo ne prepoznaje navedene tipke, umjesto tipke **[Delete]** možemo koristiti kombinaciju tipki **[Ctrl-D]**.

Nakon pokretanja programa *GNU nano* na dnu će se ekrana prikazati popis najvažnijih naredbi. Kako bi povećao prostor za unos teksta, taj se popis može ukloniti upisivanjem kombinacije tipki **[Alt-X]**. Istom se kombinacijom tipki popis može ponovno vratiti na ekran.

Interne upute za rad s programom mogu se dobiti upisivanjem kombinacije tipki **[Ctrl-G]**. Upute sadrže osnovne informacije o programu i popis svih naredbi koje se mogu koristiti. Iz uputa se u normalno radno okruženje program *GNU nano* izlazi kombinacijom tipki **[Ctrl-X]**.

```
GNU nano 1.3.12      File: torvalds.txt

Linux and Linux
-----

Linus Torvalds, a young man studying computer science at
the university of Helsinki, thought it would be a good idea
to have some sort of freely available academic version of UNIX,
and promptly started to code.

He started to ask questions, looking for answers and solutions
that would help him get UNIX on his PC. Below is one of his
first posts in comp.os.minix, dating from 1991:

From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: Gcc-1.40 and a posix-question
Message-ID: <1991Jul3.100050.9886@klaava.Helsinki.FI>
Date: 3 Jul 91 10:00:50 GMT
Hello netlanders,
```

[Read 54 lines]

```
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

Slika: Prikaz radnog ekrana programa *GNU nano*. Na dnu ekrana se vidi popis osnovnih naredbi.

5.4. Osnovne komande

Naredbe se u programu *GNU nano* zadaju odgovarajućim kombinacijama tipki.

```
[Ctrl-O] - pohranjivanje teksta u datoteku
[Ctrl-X] - završetak rada
[Ctrl-R] - učitavanje teksta iz datoteke
[Ctrl-K] - izrezivanje cijelog retka (cut)
[Ctrl-U] - umetanje izrezanog retka (paste)
[Ctrl-W] - pretraživanje teksta
[Ctrl-G] - prikazivanje uputa za rad s programom
```

Pohranjivanje teksta u datoteku

Tekst se nakon upisa ili izmjene može pohraniti u neku datoteku. Komanda za to je `[Ctrl-O]`. Nakon toga treba potvrditi ponuđeno ime datoteke ili upisati novo.

Završetak rada

Rad u programu GNU nano završava se komandom `Ctrl-X`. Ukoliko tekst na kojem smo radili nije pohranjen u datoteku, GNU nano će prije završetka rada upitati želimo li ga pohraniti u datoteku.

Učitavanje teksta iz datoteke

Tekst se pohranjen u nekoj datoteci može učitati direktno u uređivač teksta naredbom `[Ctrl-R]`. Nakon toga treba upisati ime datoteke ili odabrati je iz interaktivnog kataloga koji se pokreće komandom `[Ctrl-T]`.

Učitani tekst će biti umetnut u postojeći na mjestu na kojem se prilikom pokretanja naredbe za učitavanje iz datoteke nalazio kursor.

Izrezivanje i umetanje teksta

Komanda `[Ctrl-K]` će izrezati cijeli redak u kojem se nalazi kursor i pohraniti ga u međumemoriju. Više redaka možete pohraniti u međumemoriju višestrukim uzastopnim pritiskom na `[Ctrl-K]`.

Komanda `[Ctrl-U]` će umetnuti tekst iz međuspremnika na mjesto gdje se u tekstu nalazi kursor. Kopiranje se može ponoviti proizvoljan broj puta.

Ako se želi izrezati samo dio retka, potrebno je postaviti kursor na početak dijela koji se želi izrezati i pritisnuti `[Ctrl-^]` (ili `[Alt-A]`). Nakon toga treba pomaknuti kursor na kraj teksta koji se želi izrezati i pritisnuti `[Ctrl-K]` (slovo ispod kursora neće biti izrezano).

Kopiranje teksta može se napraviti tako da se tekst koji se želi kopirati prvo izreže, a onda se pritiskom na `[Ctrl-U]` ponovno umetne na mjesto s

kojeg je izrezan te nakon toga pozicioniranjem kursora na željenom mjestu i ponovnim pritiskom na `[Ctrl-U]` umetne tamo gdje se želi kopirati.

Pretraživanje teksta

Tekst se pretražuje tako da se pritisne `[Ctrl-W]`. Nakon toga se u statusnu liniju upiše uzorak koji se želi pronaći u tekstu. Ukoliko se traženi uzorak nalazi u tekstu, kursor će se postaviti na mjesto njegovog prvog pojavljivanja iza pozicije s koje se pokrenulo pretraživanje.

5.6. Vježba: Rad s uređivačem teksta *GNU nano*

1. U svom početnom direktoriju pokrenite program *GNU nano*.

2. Prikažite upute za rad s programom.

Upoznajte se s uputama. Možete li u uputama pronaći naredbe koje su spomenute u ovom poglavlju?

Vratite se u normalno radno okruženje.

3. Upišite sljedeći tekst:

```
Upisivanjem parametara mijenjamo nacin na koji se
izvode naredbe.
Parametri mogu biti opcije i argumenti.
Ime naredbe određuje sto treba napraviti, opcijama se
preciznije određuje kako to treba napraviti, a
argumentima se upućuje na objekt koji je predmet
obrade.
```

4. Pokušajte se kretati kroz tekst pomoću kursorskih tipki na tipkovnici.

Možete li se kretati kroz tekst pomoću kursorskih tipki? Provjerite funkcioniraju li na sustavu na kojem radite tipke `[Delete]` i `[Backspace]`.

5. Pohranite upisani tekst u datoteku s imenom `vjezba5.txt`.

6. Izadite iz programa. Provjerite nalazi li se datoteka `vjezba5.txt` u radnom direktoriju.

Koje je veličine datoteka `vjezba5.txt`? Koja su prava pristupa postavljena na tu datoteku?

7. Ponovno pokrenite program *GNU nano* navodeći kao argument datoteku `torvalds.txt` (prije pokretanja programa možete provjeriti nalazi li se navedena datoteka u Vašem radnom direktoriju).

8. Pronađite u tekstu sve riječi koje sadrže niz „Linu“.

Koje se sve riječi pojavljuju kao rezultat pretrage? Razlikuje li *GNU nano* prilikom pretrage velika od malih slova?

9. Izrežite dio teksta s citatom Torvaldsove poruke upućene na `comp.os.minix` i kopirajte ga na kraj poruke.

10. Izadite iz programa bez pohranjivanja sadržaja u datoteku.

11. Pomoću programa *GNU nano* otvorite datoteku `/etc/passwd`.

Sustav u datoteci `/etc/passwd` ima zabilježene različite podatke o korisnicima. Potražite u navedenoj datoteci liniju u kojoj se nalaze Vaši podaci (podaci o korisničkom računu koji koristite za prijavljivanje na sustav). Koje podatke prepoznajete?

12. Izadite iz programa *GNU nano*.

5.7. Pitanja za ponavljanje

1. Koja je razlika uređivača teksta i programa za obradu teksta?
2. Navedite neke uređivače teksta i neke programe za obradu teksta.
3. Kako se pokreće program *GNU nano*?
4. Kako se zadaju naredbe u programu *GNU nano*?
5. Kako se izlazi iz programa *GNU nano*?

6. Ljuska *Bash*

6.1. Što je Ljuska?

Ljuska je interpreter naredbi. Svaki put kad upišemo neku naredbu, Ljuska je preuzima, obrađuje i izvršava.

Korisnicima je na sustavima zasnovanim na *Linuxu* obično na raspolaganju više različitih ljuski. Standardna je Ljuska ***Bash*** (*Bourne Again Shell*).

Ljuske se međusobno razlikuju po sintaksi ugrađenog interpreterskog jezika, opcijama koje podržavaju, konfiguracijskim datotekama i drugim detaljima.

6.2. Upis naredbi

Ljuska *Bash* standardno sve što se upiše u jedan redak smatra jednom naredbom.

Ako želimo više naredbi upisati u jedan redak, tada ih trebamo odvojiti znakom `;` (točka-zarez).

Ako jednu naredbu želimo napisati u više redaka tada na kraj retka trebamo upisati znak `\` (*backslash*).

Prilikom pisanja naredbi u više redova, nakon prvog reda pojaviti će se **sekundarni prompt**. To je obično znak `>` (veće) .

Primjeri:

```
$ echo Jedan ; echo Dva
Jedan
Dva
$ echo Jedan \
> Dva
Jedan Dva
$
```

6.3. Variable

Ljuska *Bash* omogućava rad s varijablama.

Varijabla je ime koje predstavlja neku vrijednost. Vrijednost varijable se može mijenjati.

Imena varijabli obično se pišu velikim slovima.

Vrijednosti koje mogu poprimiti varijable su nizovi zakova, a pridružuju im se pomoću operatora `=` (sintaksa: `VARIJABLA=VRIJEDNOST`).

Prilikom pridruživanja vrijednosti varijablama, ispred znaka jednakosti ne smije se nalaziti niti jedno prazno mjesto (znak `[Space]`).

Siguran način pisanja vrijednosti koje se pridružuju varijablama je korištenjem navodnika (bilo jednostrukih, bilo dvostrukih).

Primjeri ispravnog i neispravnog pridruživanja vrijednosti varijablama:

ispravno	neispravno
ABC="123"	ABC= "123"
ABC=123	ABC =123
POZDRAV="Dobar dan"	POZDRAV=Dobar dan

Podrazumijevana inicijalna vrijednost svih varijabli je prazan niz (NULL).

Sustav koristi neke varijable u svoje svrhe. Vrijednosti tih varijabli se postavljaju prilikom prijavljivanja korisnika.

Vrijednost varijable poziva se tako da se u naredbi ispred imena varijable napiše znak \$. U tom slučaju ljuska na mjesto gdje se nalazi ime varijable umeće njezinu vrijednost. **Taj se postupak događa prije izvršavanja naredbe.**

Jednostavan i uobičajen način ispisivanja vrijednosti varijable je pomoću naredbe `echo`.

Primjer:

```
$ ABC=123
$ echo $ABC
123
$
```

Postoje dvije vrste varijabli: lokalne varijable i varijable okruženja.

6.3. Lokalne varijable

Lokalne varijable dostupne su samo u ljusci u kojoj su stvorene.

Ukoliko se drugačije ne naznači, svaka je varijabla lokalna.

Naredba **set** ispisuje popis svih koje su joj dostupne zajedno s njihovim vrijednostima.

Sintaksa:

set

Primjer:

```
$ ABC=123
$ echo $ABC
123
$ bash          ← pokrećemo novu ljusku (ljuska je također program)
$ echo $ABC
                  ← u novoj ljusci vrijednost varijable ABC nije postavljena
$ exit          ← vraćamo se u polaznu ljusku
$ echo $ABC
123
$
```

Napomena

Vrijednosti lokalnih varijabli nisu dostupne programima koje ljuska pokreće.

6.4. Varijable okruženja i naredba `export`

Vrijednosti **varijabli okruženja** dostupne su programima koje pokreće ljuška.

Naredba `export` pretvara lokalne varijable u varijable okruženja. Taj postupak nazivamo **izvozom varijabli**.

Ako se naredba `export` pokrene bez argumenata, ispisat će se popis varijabli koje su izvezene u aktualnoj ljušci.

Popis svih varijabli okruženja ispisuje naredba `env`. Taj popis čine varijable koje su izvezene u aktualnoj ljušci, ali i varijable okruženja koje je aktualna ljuška naslijedila.

Sintaksa:

```
export [VARIJABLA]
env
```

Primjer:

```
$ ABC=123
$ echo $ABC
123
$ export ABC      ← izvozimo varijablu ABC
$ bash           ← pokrećemo novu ljušku
$ echo $ABC
123              ← ovaj je put vrijednost varijable ABC dostupna
$               i u novoj ljušci
```

Napomena

Iz ljuške izlazimo naredbom `exit` ili kontrolnim znakom `[Ctrl-D]`.

6.5. Vrste naredbi

Naredbe mogu biti:

- ugrađene u ljušku
- izvršne datoteke
- aliasi.

Prilikom pokretanja neke naredbe prvo što ljuška radi je otkrivanje tipa naredbe. Ukoliko se radi o **izvršnoj datoteci**, ljuška je treba pronaći na disku kako bi je mogla pročitati i izvršiti. Da bi ljuška mogla pronaći naredbu koja je izvršna datoteka, naredba mora biti zadana stazom koja vodi do nje ili se mora nalaziti u direktoriju koji je naveden u varijabli `PATH`.

Vrijednost varijable `PATH` čini niz imena direktorija koji su međusobno odijeljeni znakom `:` (dvotočka).

Primjer:

```
$ echo $PATH
/home/tecaj00/bin:/usr/local/bin:/usr/bin:/bin:.
```

↑
uočite da je u ovom primjeru dio popisa direktorija u kojima ljuška provjerava postoji li zadana naredba direktorij `.` (radni direktorij)

Aliasi su naredbe koje su definirane kao sinonimi ili zamjenska imena za neke druge naredbe. Naredba **alias** ispisuje popis aliasa i omogućava njihovo postavljanje.

Sintaksa:

```
alias [ALIAS=NAREDBA]
```

Primjer:

```
$ alias                                ← ispisujemo popis aliasa
alias md='mkdir'
alias rd='rmdir'                      ← u ovom su nam trenutku dostupna dva
                                     aliasa: md i rm

$ moj_ls                              ← pokušavamo pokrenuti naredbu moj_ls
-bash: moj_ls: command not found
      ↑
    sustav nam javlja da naredba moj_ls ne postoji

$ alias moj_ls='ls -a'
      ↑
    definiramo alias moj_ls kao ls -a

$ moj_ls                              ← ponovno pokušavamo pokrenuti moj_ls
.bash_history a b c
      ↑
    ovaj smo put dobili smo rezultat koji smo željeli

$
```

Naredba **type** ispisuje kojeg je tipa naredba.

Sintaksa:

```
type NAREDBA
```

Primjer:

```
$ type pwd
pwd is a shell builtin                ← pwd je naredba ugrađena u
                                     ljusku

$ type cat
cat is /bin/cat                       ← cat je izvršna datoteka na
                                     disku (u direktoriju /bin)

$ type rd
rd is aliased to 'rmdir'              ← rd je alias (kratica za rmdir)
```

Naredba **which** ispisuje stazu do naredbi koje su izvršne datoteke.

Da bi naredba **which** mogla pronaći zadanu naredbu na disku, potrebno je da se ona nalazi u direktoriju koji je naveden u varijabli **PATH**.

Sintaksa:

```
which NAREDBA
```

Primjer:

```
$ which info
/usr/bin/info
$
```

6.6. Vježba: Varijable i naredbe**1. Proučite vrijednost varijable `PATH`.**

Provjerite nalazi li se `.` u popisu direktorija? Na kojem je mjestu?

2. U svom početnom direktoriju stvorite poddirektorij poglavlje6. U novostvorenom direktoriju stvorite datoteku `test`. Dodijelite si pravo izvršavanja te datoteke. Neka Vaš radni direktorij bude direktorij `poglavlje6`. Pokrenite naredbu `which test`.

Što se dogodilo?

3. Neka Vaš radni direktorij ponovno bude Vaš početni direktorij. Ponovno pokrenite naredbu `which test`.

Je li i ovaj put naredba `which` uspjela pronaći izvršnu datoteku `test`?

4. Stvorite alias `dir` koji će ispisivati sadržaj direktorija (definirajte taj alias kao naredbu `ls` s opcijama koje smatrate korisnima).

Provjerite radi li alias. Pronađite navedeni alias u popisu svih aliasa.

5. Provjerite jesu li varijable `PATH` i `MAIL` varijable okruženja.**6. Dodijelite varijabli `MOJA_NAREDBA` vrijednost `ls` (`MOJA_NAREDBA=ls`). Kao novu naredbu napišite `$MOJA_NAREDBA`.**

Što se dogodilo? Je li to očekivani učinak?

7. Koja je vrijednost varijable `PS1`? Promijenite vrijednost varijable `PS1` tako da ona bude „`[\u@\h \w]\$`“ (upišite `PS1="[\u@\h \w]\$`“).

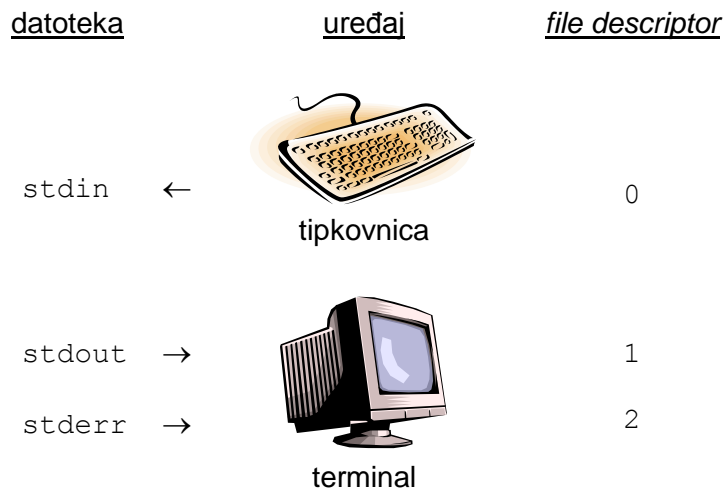
Što se dogodilo? Kako biste promptu vratili stari izgled?

6.7. Pitanja za ponavljanje

1. Što je ljuska? Koja je standardna ljuska na *Linuxu*?
2. Kako možemo napisati dvije naredbe u jednoj liniji?
3. Što se pojavljuje u novoj liniji kad u prethodnoj ne završimo s upisom naredbe?
4. Koja je razlika između lokalnih varijabli i varijabli okruženja? Kako lokalne varijable pretvaramo u varijable okruženja?
5. Kako pozivamo vrijednost varijable?
6. Koji su tipovi naredbi? Kako se može doznati kojeg je naredba tipa?
7. Čemu služi varijabla `PATH`?
8. Što je alias?

7. Preusmjeravanje ulaza i izlaza naredbi

7.1. `stdin`, `stdout` i `stderr`



Prilikom pokretanja ljuske automatski se stvaraju tri datoteke:

- standardni ulaz (`stdin`)
- standardni izlaz (`stdout`)
- standardni izlaz za greške (`stderr`)

Standardni ulaz je datoteka iz koje ljuska čita (obično je ta datoteka preusmjerena na tipkovnicu).

Standardni izlaz je datoteka u koju ljuska ispisuje poruke (obično je to ekran).

Standardni izlaz za greške je datoteka u koju ljuska ispisuje poruke o greškama (a to je obično također ekran).

Primjer poruke o greški je:

```
-bash: moj_ls: command not found.
```

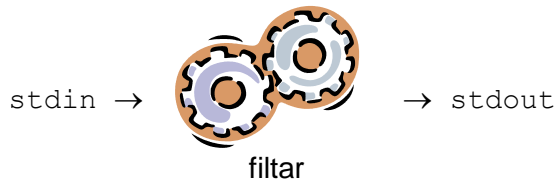
(S ovom smo se porukom susreli u jednom od primjera u prethodnom poglavlju.)

Svaka naredba koja se pokrene također ima svoj standardni ulaz, standardni izlaz i standardni izlaz za greške. Ukoliko se ne zada drugačije, oni su preusmjereni na standardni ulaz, standardni izlaz i standardni izlaz od ljuske.

U ovom ćemo poglavlju upoznati mehanizme za preusmjeravanje ulaza i izlaza i njihovu primjenu.

7.2. Filtri

Filtri su programi koji čitaju podatke sa standardnog ulaza, obrađuju ih, a rezultate ispisuju na standardni izlaz.



Neki od programa koje smo upoznali u prethodnim poglavljima mogu raditi kao filtri.

Primjer:

Da bi naredba `cut` radila kao filter dovoljno je pokrenuti je bez argumenata. U tom će slučaju naredba `cut` umjesto iz neke datoteke podatke čitati sa standardnog ulaza.

```

$ cat          ← pokrećemo naredbu cat bez argumenata
Jedan         ← upisujemo prvi redak teksta
Jedan         ← naredba cat ispisuje ono što smo upisali
Dva           ← ponavljamo prethodni korak
Dva
[Ctrl-D]      ← upis završavamo s [Ctrl-D]
$
  
```

Napomena

Znak `[Ctrl-D]` je na Linuxu oznaka za kraj datoteke (EOF).

Sljedeća tablica opisuje neke od filtra:

Ime naredbe	Opis	Sintaksa
sort	sortira ulazne podatke	<code>sort [-n] [+POLJE] [DATOTEKA]</code>
grep	ispisuje samo linije koje sadrže zadani uzorak	<code>grep UZORAK [DATOTEKA]</code>
wc	broji linije, riječi i znakove	<code>wc [-lwc] [DATOTEKA]</code>

Ako se gore navedeni programi pokrenu bez zadavanja datoteke, oni će raditi kao filtri. Ako se prilikom pokretanja tih naredbi navede ime datoteke, tada će one čitati podatke iz nje (neće raditi kao filtri).

7.3. Preusmjeravanje ulaza: <

Ljuska *Bash* omogućava da prilikom pokretanja bilo koje naredbe preusmjerimo njezin standardni ulaz.

Preusmjeravanje zadajemo operatorom `<` iza kojeg treba navesti ime datoteke koja se povezuje sa standardnim ulazom. Operator i ime datoteke treba navesti iza naredbe.

Sintaksa:

NAREDBA < DATOTEKA

Preusmjeravanje standardnog ulaza znači, na primjer, da će naredba umjesto s tipkovnice podatke učitavati iz neke datoteke. Pri tome naredba ne zna da je došlo do preusmjeravanja. Preusmjeravanje je za naredbu nevidljivo i obavlja ga ljuska.

Jedan od mehanizama koji omogućavaju ovu funkcionalnost je činjenica koja je spomenuta u poglavlju 3. *Sustav datoteka*: hardverski uređaji su na *Linuxu* predstavljeni kao datoteke. To znači da programi na isti način mogu pristupiti običnim datotekama i hardverskim uređajima.

Primjer (pokretanje naredbe `cat` s preusmjeravanjem ulaza):

```
$ cat < /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
...
$
```

Ovaj primjer pokazuje već treći način uporabe naredbe `cat`:

1. s navođenjem argumenata: `cat DATOTEKA`
2. bez argumenata i bez preusmjeravanja ulaza: `cat`
3. bez argumenta, ali s preusmjeravanjem ulaza: `cat < DATOTEKA`.

U drugom i trećem slučaju naredba `cat` radi kao filter.

Vrijedi uočiti da ćemo, ako napišemo

```
cat /etc/passwd ili cat < /etc/passwd
```

dobiti isti rezultat na ekranu. Ipak, premda će za krajnjeg korisnika rezultat biti isti, način na koji će naredba `cat` doći do rezultata je bitno različit.

U prvom će se slučaju (`cat /etc/passwd`) dogoditi sljedeće:

1. Ljuska će pokrenuti naredbu `cat`.
2. Naredba `cat` će provjeriti je li zadana s nekim argumentom.
3. Naredba `cat` će pokušati otvoriti datoteku čije ime odgovara zadanom argumentu.
4. U slučaju uspješnog otvaranja datoteke, naredba `cat` će čitati podatke iz nje i ispisivati ih na standardni izlaz (na primjer, na ekran). U slučaju neuspješnog pokušaja otvaranja datoteke ispisat će poruku o greški.
5. Kad naiđe na kraj datoteke, naredba `cat` će je zatvoriti i prekinuti sa izvođenjem.

U tom slučaju naredba `cat` sama upravlja pristupom datoteci, ispisuje eventualnu poruku o greški i slično.

U drugom slučaju (`cat < /etc/passwd`) dogodit će se sljedeće:

1. Ljuska će izvršiti preusmjeravanje ulaza i pokrenut će naredbu `cat`.
2. Naredba `cat` će provjeriti je li zadana s nekim argumentom.
3. S obzirom na to da argument nije zadan, naredba `cat` će čitati podatke sa standardnog ulaza i ispisivati učitano na standardni izlaz.
4. Kad naiđe na znak za kraj datoteke (EOF), naredba `cat` će prekinuti s izvođenjem.

U ovom slučaju, svu brigu oko rada s datotekom obavlja ljuska.

Proučimo primjer koji to dokazuje:

```
$ cat /nema_me
cat: /nema_me: No such file or directory
$ cat < /nema_me
-bash: /nema_me: No such file or directory
$
```

Proučimo detaljnije što se dogodilo u gornjem primjeru:

```
$ cat /nema_me      ← kao argument smo zadali ime datoteke
                    koja ne postoji
```

```
cat: /nema_me: No such file or directory
```

↑

poruku o greški ispisuje naredba `cat`.

```
$ cat < /nema_me    ← pokrećemo naredbu bez argumenata,
                    s preusmjeravanjem ulaza na datoteku
                    koja ne postoji
```

```
-bash: /nema_me: No such file or directory
```

↑

poruku o greški ispisuje ljuska, a **naredba `cat` uopće nije pokrenuta**

Sljedeća tablica usporedno prikazuje različite načine pokretanja naredbe `cat`, ovisno o načinu čitanja podataka koje treba ispisati:

Izvođenje programa uz čitanje sa <code>stdin</code>	Izvođenje programa uz zadani argument	Izvođenje programa s preusmjeravanjem standardnog ulaza
<pre>\$ cat</pre> <p>upisujemo tekst, naredba ga ispisuje za kraj upisujemo [Ctrl-D].</p>	<pre>\$ cat datoteka</pre> <p>ispisuje se sadržaj datoteke</p>	<pre>cat < datoteka</pre> <p>ispisuje se sadržaj datoteke</p>

7.4. Preusmjeravanje izlaza: > i >>

Ljuska *Bash* omogućava preusmjeravanje standardnog izlaza.

Preusmjeravanje standardnog izlaza moguće je zadati operatorima > i >>.

Sintaksa:

```
NAREDBA > DATOTEKA
NAREDBA >> DATOTEKA
```

U prvom slučaju, ljuska će otvoriti **novu** datoteku sa zadanim imenom, a u drugom će slučaju standardni izlaz preusmjeriti na kraj već **postojeće** datoteke (u datoteku će se nadopisivati). Ako u drugom slučaju navedena datoteka ne postoji, otvorit će se nova datoteka.

Primjeri:

```
$ echo Jedan > ispis.dat
$ cat ispis.dat
Jedan
$ echo Dva >> ispis.dat
$ cat ispis.dat
Jedan
Dva
$ echo Tri > ispis.dat
$ cat ispis.dat
Tri
$
```

7.5. Preusmjeravanje standardnog izlaza za poruke o greškama: 2> i 2>>

Preusmjeravanje standardnog izlaza za poruke o greškama vrlo je slično preusmjeravanju standardnog izlaza.

Operatori kojima zadajemo preusmjeravanje su:

```
2>      - preusmjeravanje u novu datoteku
2>>     - preusmjeravanje na kraj postojeće datoteke.
```

Sintaksa:

```
NAREDBA > DATOTEKA
NAREDBA >> DATOTEKA
```

Primjeri:

```
$ cp
cp: missing file argument
Try `cp --help' for more information.
$ cp 2> greska.txt
$ cat greska.txt
cp: missing file argument
Try `cp --help' for more information.
$
```

Preusmjeravanje standardnih ulaza i izlaza moguće je koristiti istovremeno:

```
$ sort < popis_nesortiran > popis_sortiran
$ moj_program < podaci > rjesenje 2> greske
```

7.6. Ulančavanje

Jedna od mogućih primjena preusmjeravanja standardnih ulaza i izlaza je pohranjivanje ispisa naredbe u datoteku i uporaba te datoteke kao standardnog ulaza druge naredbe.

Sljedeći primjer prebrojavanja broja datoteka i direktorija u radnom direktoriju to pokazuje:

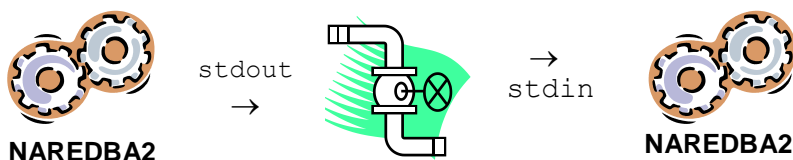
```
$ ls > /tmp/privremena_datoteka
$ wc -w < /tmp/privremena_datoteka
18
$
```

Nedostatak ovog pristupa je stvaranje privremene datoteke. To se može izbjeći uporabom mehanizma ulančavanja naredbi.

Ulančavanje naredbi je preusmjeravanje standardnog ulaza jedne naredbe na standardni izlaz druge naredbe. Ulančavanje se zadaje operatorom `|` (*pipe*).

Sintaksa:

NAREDBA1 | NAREDBA2



Sljedeći primjer je rješenje za gore navedeni problem prebrojavanja broja datoteka i direktorija uporabom mehanizma ulančavanja:

```
$ ls | wc -w
18
$
```

Primjeri:

```
$ who | wc -l
$ who | grep tecaj | wc -l
```

Ulančavanje se vrlo često koristi s naredbama `more` i `sort`:

```
$ ls -lR | more    ← rekurzivno ispisujemo sadržaj direktorija,
                   ekran po ekran

$ ls | sort        ← ispisujemo sortirani popis datoteka
```

7.7. Vježba: Preusmjeravanje ulaza i izlaza, filtri i ulančavanje naredbi

1. Izlučite poruke o greškama koje ispisuje naredba `ls -lR /etc`.

Koje su to poruke o greškama? Zašto ih sustav ispisuje?

Uputa: Koristite preusmjeravanje standardnog izlaza i standardnog izlaza za greške)

2. Za koliko varijabli ljska u kojoj radite ima postavljene vrijednosti? Koliko je od toga varijabli okruženja? Koliko je varijabli okruženja eksportirano od strane ljske u kojoj radite?

Razlikuje li se odgovor na posljednja dva pitanja? Zašto?

(Za prebrojavanje varijabli koristite naredbu `wc`.)

3. Pomoću naredbe `grep` ispišite na ekran liniju koja u datoteci `/etc/passwd` opisuje Vaš korisnički račun.

Na koliko načina možete riješiti ovaj zadatak?

4. Koliko je na sustavu korisničkih računa namijenjenih polaznicima tečajeva?

Uputa: Imena korisničkih računa koji su namijenjeni polaznicima tečajeva počinju s `tecaj`.

7.8. Pitanja za ponavljanje

1. Što su `stdin`, `stdout` i `stderr`?
2. Što su filtri?
3. Navedite nekoliko filtara.
4. Kako se preusmjerava standardni ulaz naredbi?
5. Koji su vrste preusmjeravanja standardnog izlaza naredbi?
6. Čemu služi standardni izlaz za greške?
7. Kako se preusmjerava standardni izlaz za greške?
8. Koji je razlika između `cat DATOTEKA` i `cat < DATOTEKA`?
9. Što je ulančavanje?

8. Procesi

8.1. Općenito o procesima

Procesi su programi koji se izvršavaju.

Programi su izvršene datoteke.

Prilikom pokretanja sustav svakom procesu dodjeljuje njegov broj (**PID**). Taj je broj jedinstven. Sustav također uz svaki proces bilježi i broj procesa koji je pokrenuo (**PPID**), kao i njegov radni direktorij.

8.1. Popis procesa: **ps** i **top**

Naredba **ps** ispisuje popis procesa na sustavu.

Sintaksa:

```
ps [-f] [-e | -u KORISNIK]
top
```

Osnovne opcije su sljedeće:

- e** - ispis svih procesa na sustavu (u suprotnom će naredba **ps** ispisati samo procese koji su vezani uz ljusku u kojoj se radi)
- f** - ispis popisa s dodatnim podacima o procesima.

Primjeri:

```
$ ps
  PID TTY          TIME CMD
 13092 pts/1        00:00:00 bash
 13120 pts/1        00:00:00 ps
$ ps -f
UID          PID    PPID  C STIME TTY          TIME CMD
tecaj01    13092   13091  0 09:20 pts/1        00:00:00 -bash
tecaj01    13121   13092  0 09:20 pts/1        00:00:00 ps -f
```

Naredba **top** na ekranu kontinuirano ispisuje popis procesa poredanih prema količini procesorskog vremena koji troše.

Sintaksa:

```
top
```

Na mnogim je sustavima onemogućeno da korisnici bez administratorskih korisničkih prava imaju uvid u popis svih procesa na sustavu. U tom će slučaju naredbe **ps** (tj. **ps -e**) i **top** ispisivati samo popise procesa koji pripadaju korisniku koji je naredbu pokrenuo.

8.2. Izvršavanje naredbi u pozadini

Do sada smo naredbe zadavali tako da smo nakon njenog pokretanja morali čekati da sustav završi s njenim izvođenjem da bismo mogli zadati sljedeću. Takav način rada se zove **izvršavanje naredbi u prednjem planu**.

Pri tome je pojavljivanje prompta služilo, između ostaloga, kao znak je li sustav završio s izvršavanjem prethodne naredbe.

Ljuska omogućava da izbjegnemo ograničenje istovremenog pokretanja samo jedne naredbe. Postupak koji možemo u tu svrhu koristiti je **izvršavanje naredbi u pozadini**.

Izvršavanje naredbi u pozadini zadaje se znakom **&** na kraju linije s naredbom.

Sintaksa:

NAREDBA &

Nakon pokretanja naredbe s izvršavanjem u pozadini, ljuska će ispisati njezin PID kao znak da će se izvođenje nastaviti u pozadini, a odmah nakon toga pojaviti će se prompt i moći ćemo nastaviti s radom.

Po završetku izvođenja naredbe u pozadini, ispisat će se odgovarajuća poruka.

Primjer:

```
$ ls -lR > /tmp/popis &      ← pokrećemo naredbu u pozadini
[1] 16423                    ← PID je 16423, a redni broj koji
                             je ljuska dodijelila procesu je 1

$                             ← pojavio se prompt i možemo
                             nastaviti s radom

...
[1]+  Done    ls -lR >/tmp/popis
    ↑
    na kraju dobivamo poruku da je izvršavanje naredbe
    u pozadini završilo
```

8.4. Upravljanje procesima u ljusci

fg	- prebacivanje procesa u prednji plan
bg	- nastavak izvršavanja zaustavljenog procesa u pozadini
[Ctrl-Z]	- zaustavljanje procesa koji se izvršava u prednjem planu
jobs	- popis procesa koje se izvršavaju u pozadini

Izvršavanje naredbi koje smo pokrenuli u prednjem planu, možemo privremeno zaustaviti utipkavanjem kontrolnog znaka `[Ctrl-Z]`. Ljuska će ispisati odgovarajuću poruku i ispisati prompt. Nakon toga možemo zadavati naredbe ljusci. Zaustavljeni program ostaje u memoriji računala i čeka da zadamo neku od naredbi za nastavak njegovog izvođenja.

Izvođenje naredbi koje su na takav način zaustavljene možemo:

- nastaviti izvoditi u pozadini (naredba `bg`)
- nastaviti izvoditi u prednjem planu (naredba `fg`).

Naredba `fg` nam omogućava prenošenje u prednji plan ne samo procesa koji su zaustavljeni sa `[Ctrl-Z]`, nego i svih koji se izvršavaju u pozadini.

Naredba `jobs` ispisuje popis svih procesa koji se izvršavaju u pozadini.

Primjer:

```
$ ls -lR > popis ← naredbu pokrećemo u prednjem planu
...
[Ctrl-Z] ← zaustavljamo njeno izvođenje sa [Ctrl-Z]
[1]+  Stopped                  ls -lR > popis
    ↑
    ispisuje se poruka, prvi podatak u poruci je redni
    broj koji je procesu dodijelila ljuska

$
    ← pojavio se prompt i možemo
    nastaviti s radom

...
$ bg ← zadajemo nastavak izvođenja naredbe
    u pozadini
[1]+ ls -lR > popis &
    ↑
    ljuska nas obavještava da je izvršavanje naredbe
    nastavljeno u pozadini
```

Napomena

U većini slučajeva izvođenje naredbe u prednjem planu možemo prekinuti utipkavanjem kontrolnog znaka `[Ctrl-C]`.

Napomena

Važno je razlikovati prekidanje od zaustavljanja izvođenja.

Prekidanje izvođenja znači da je izvođenje naredbe prestalo, da je naredba obrisana iz memorije računala i da se njeno izvođenje više ne može nastaviti.

Prekid izvođenja je konačan, dok je zaustavljanje izvođenja privremeno.

8.3. Signali i naredba `kill`

Linux omogućava slanje signala procesima naredbom `kill`.

Procesima se mogu slati različiti signali, različitih značenja. Neki od signala su:

- **SIGHUP** (1) - zahtjev za re-inicijalizacijom procesa
- **SIGKILL** (9) - zahtjev za grubim prekidom izvršavanja
- **SIGTERM** (15) - zahtjev za prekidom izvršavanja (softverski završetak rada)

Sintaksa:

```
kill [-s SIGNAL] PROCES
```

Napomena

Programi mogu biti napisani tako da ignoriraju primljene signale (odnosno tako da sami određuju svoje ponašanje u slučaju da prime signal).

Izuzetak je signal `SIGKILL` (9). Njega programi ne mogu ignorirati.

Naredbi `kill` se kao argument treba zadati PID procesa čije izvođenje želimo prekinuti ili redni broj koji je procesu dodijelila ljuska. Ako se navodi redni broj koji je procesu dodijelila ljuska tada ispred njega treba upisati znak `%` (postotak).

Ako se ne navede signal koji šaljemo, podrazumijeva se da se radi o signalu `SIGTERM`. Signale možemo zadati njihovim rednim brojem ili simboličkim imenom.

Primjeri:

```
$ jobs
[1]-  Running                  program1 &
[2]+  Running                  vjezba8 &
```

↑

u ovom se primjeru u pozadini izvršavaju dva procesa

```
$ ps
  PID TTY          TIME CMD
15028 pts/0    00:00:00 bash
29847 pts/0    00:00:00 program1
29281 pts/0    00:00:00 vjezba8
24783 pts/0    00:00:00 ps
```

← njihove PID-ove možemo doznati naredbom `ps`

```
$ kill -9 29847
```

← *program1 prekidamo navodeći njegov PID*

```
[1]+  Killed                  program1
```

```
$ kill -9 %2
```

← *program vjezba2 prekidamo navodeći redni broj dodijeljen od ljuske*

```
[2]+  Killed                  vjezba8
$
```

8.4. Vježba: Upravljanje procesima

1. U poddirektoriju poglavlje8 nalaze se dva programa: `brzi` i `spori`. Pokrenite ih.

Što rade navedeni programi?

2. Pokrenite program `spori` u pozadini (upišite `spori &`).

Možete li nastaviti raditi u ljusci? Što se događa s podacima koje ispisuje program `spori`? Ometa li Vas u radu ispis tih podataka? Kako biste riješili taj problem?

3. Pričekajte da završi izvršavanje programa `spori`. Ponovno pokrenite program `spori`, ovaj put u prednjem planu. Zaustavite njegovo izvršavanje.

4. Pokrenite naredbu `jobs`.

Koliko se procesa nalazi u pozadini? Koji je njihov status?

5. Zadaajte nastavak izvršavanja zaustavljenog programa u pozadini.**6. Ponovno pokrenite naredbu `jobs`.**

Je li se promijenio status programa u pozadini?

7. Koji je PID programa koji se nalazi u pozadini?**8. Zaustavite izvođenje programa pomoću naredbe `kill`.**

Uputa: Ukoliko je program `spori` u međuvremenu završio s izvođenjem, pokrenite ga ponovno kako biste mogli izvršiti navedeni zadatak.

9. Ponovno pokrenite program `spori` (u prednjem planu). Prekinite izvođenje programa utipkavanjem kontrolnog znaka `[Ctrl-C]`.

Jeste li uspjeli prekinuti izvođenje programa?

10. Ponovno pokrenite program `spori`. Zaustavite ga. Provjerite nalazi li se zaustavljeni program na popisu koji daje naredba `jobs`.**11. Upišite `kill -9` navodeći kao argument PID ljsuke u kojoj radite.**

Što se dogodilo?

Uputa: PID ljsuke možete doznati pomoću naredbe `ps`. Neka je PID Vaše ljsuke 24454 kao u sljedećem primjeru:

```
$ ps
  PID TTY          TIME CMD
 15028 pts/0    00:00:00 bash
 24454 pts/0    00:00:00 ps
$ kill -9 24454
```

12. Ponovno se prijavite na sustav. Provjerite nalazi li se zaustavljeni program i dalje na popisu koji daje naredba `jobs`.

Koji je razlog što programa više nema na popisu?

Napomena

Ako želimo da neki program nastavi raditi i nakon što se odjavimo sa sustava, tada je potrebno taj program pokrenuti pomoću naredbe `nohup`.

Na primjer:

```
nohup obrada > rezultati &
```

8.5. Pitanja za ponavljanje

1. Što su procesi?
2. Čemu služi naredba `ps`?
3. Kako se zadaje izvršavanje naredbi u pozadini?
4. Kako se zaustavlja izvođenje naredbi?
5. Što se može napraviti sa zaustavljenim naredbama?
6. Kako se prekida izvođenje naredbi u prednjem planu?
7. Kako se prekida izvođenje naredbi u pozadini?

9. Arhiviranje i sažimanje datoteka

9.1. Arhiviranje datoteka: tar

Naredba `tar` omogućava izradu, pregledavanje i raspakiravanje arhiva. Arhive su datoteke koje u sebi sadrže zapakirane druge datoteke.

Sintaksa:

```
tar c[v]f ARHIVA POPIS
tar t[v]f ARHIVA
tar x[v]f ARHIVA [POPIS]
```

Ponašanje naredbe `tar` ovisi o ključu – prvoj zadanoj opciji, koja može biti:

- c** - izrada arhive
- t** - pregledavanje sadržaja arhive
- x** - raspakiravanje cijele ili dijela arhive

Opcijom **f** se označava da se arhiviranje vrši u datoteku i da je sljedeći argument ime datoteke u koju se arhivira. (Ako se ta opcija ne navede, arhiviranje se radi na standardni izlaz.)

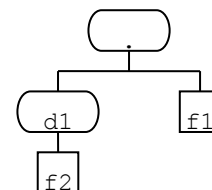
Ako se navede opcija **v**, tada naredba `tar` tijekom rada ispisuje dodatne podatke (npr. popis datoteka koje se arhiviraju ili raspakiravaju).

Obično se arhivama načinjenim naredbom `tar` daju imena s nastavkom `.tar` (npr. `moji_podaci.tar`).

Primjeri:

```
$ ls -RF                                ← naredbom ls -RF ispisujemo sadržaj
.:                                       radnog direktorija:
d1/  f1
```

```
./d1:
f2
```



```
$ tar cvf /tmp/arhiva.tar .            ← pokrećemo izradu arhive
./                                       u datoteku arhiva.tar
./d1/
```

```
./d1/f2                                ← zadana je opcija c, pa se ispisuje popis
./f1                                   arhiviranih datoteka i direktorija
```

```
$ tar tf /tmp/arhiva.tar                ← provjeravamo što se
./                                       nalazi u arhivi
./d1/
```

```
./d1/f2
./f1
```

```
$ mkdir /tmp/vjezba                    ← stvaramo direktorij u koji ćemo
                                        raspakirati dio arhive i postavljamo
$ cd /tmp/vjezba                       taj direktorij za radni direktorij
```

```
$ ls -l
total 0
```

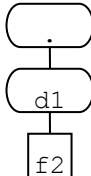
← *novi direktorij je prazan*

```
$ tar xvf /tmp/arhiva.tar ./d1
./d1/
./d1/f2
```

↑
iz arhive raspakiravamo poddirektorij d1 i njegov sadržaj (datoteku f2)

```
$ ls -FR
.:
d1/
./d1:
f2
$
```

← *provjeravamo što je raspakirano iz arhive:*



```

graph TD
    A((.)) --> B((d1))
    B --> C[f2]
  
```

9.2. Sažimanje datoteka: gzip

Datoteke možemo sažimati pomoću naredbe **gzip**. Tom naredbom, uz navođenje dodatnih opcija, možemo pregledavati sadržaj tako nastalih datoteka (opcija **-l**), te raspakirati ih (opcija **-d**).

Prilikom sažimanja neke datoteke, nastat će nova datoteka s istim imenom uz dodatak nastavka **.gz**. Originalna datoteka bit će obrisana.

Sve opcije naredbe **gzip** moguće je dobiti navođenjem opcije **-h**.

Sintaksa:

```
gzip DATOTEKA
gzip -l DATOTEKA
gzip -d DATOTEKA
```

Primjeri:

```
$ ls -l a.tar
-rw-r--r-- 1 tecaj01 tecaj 10240 Oct 23 18:02 a.tar
```

↑
veličina datoteke a.tar prije sažimanja iznosi 10 KB

```
$ gzip a.tar
```

← *datoteku sažimamo pomoću naredbe gzip*

```
$ ls -l a.tar*
-rw-r--r-- 1 tecaj01 tecaj 181 Oct 23 18:02 a.tar.gz
```

↑
veličina datoteke a.tar.gz koja je nastala nakon sažimanja iznosi samo 181 bajt

```
$ gzip -l a.tar.gz
compressed  uncompressed  ratio  uncompressed_name
      181           10240   98.5%      a.tar
```

↑
naredba gzip -l potvrđuje raniju opasku o visokom stupnju sažimanja

```
$ gzip -d a.tar.gz      ← datoteku vraćamo u prvobitno stanje
$                       pomoću naredbe gzip -d
```

9.3. Vježba: Arhiviranje i sažimanje datoteka

1. Neka Vaš radni direktorij bude direktorij `poglavlje3`.

Koji je sadržaj tog direktorija? (Riječ je o direktoriju koji se koristio u jednoj od prethodnih vježbi.)

2. Arhivirajte sadržaj radnog direktorija u datoteku `~/arhiva.tar`.

Gdje se nalazi navedena datoteka? Koja je njezina veličina? Zabilježite taj podatak. Koje se datoteke nalaze u arhivi?

3. Sažmite datoteku `~/arhiva.tar` pomoću naredbe `gzip`.

Što se dogodilo s originalnom datotekom?

4. Uklonite datoteku `~/arhiva.tar.gz` sa sustava.

9.4. Pitanja za ponavljanje

1. Čemu služi naredba `tar`? Navedite neki primjer njene praktične primjene.
2. Čemu služi naredba `gzip`?

10. Linux na mreži

10.1. Pristup udaljenim sustavima: ssh i scp

Naredba **ssh** omogućava siguran terminalski rad na udaljenim sustavima.

Sintaksa:

```
ssh [KORISNIK@] IME_SUSTAVA
```

Primjer:

```
$ ssh tecaj01@baltazar.srce.hr
      ↑
    upisujemo naredbu na lokalnom sustavu
Password:
...
$ ← prompt na udaljenom sustavu
```

Naredba **scp** omogućava siguran prijenos (kopiranje) datoteka između lokalnog i udaljenog sustava.

Sintaksa:

```
ssh IZVOR DESTINACIJA
```

IZVOR i DESTINACIJA su oblika:

```
[ [KORISNIK@] IME_SUSTAVA: ] DATOTEKA
```

Primjer:

```
kopiramo datoteku a.txt iz polaznog direktorija korisnika
tecaj01 na sustavu baltazar.srce.hr (služeći se pri tome
korisničkim računom i ovlastima korisnika tecaj01) u radni
direktorij na lokalnom sustavu dodjeljujući pri tome novoj datoteci
ime b.txt
      ↓
$ scp tecaj01@baltazar.srce.hr:a.txt b.txt
Password:
a.txt      100%  502      0.5KB/s   00:00
$
```

10.2 Mrežno ime i domena sustava: hostname i dnsdomainname

Naredba **hostname** ispisuje mrežno ime sustava na kojem radimo.

Naredba **dnsdomainname** ispisuje ime domene kojoj pripada sustav.

Sintaksa:

```
hostname
```

dnsdomainname**Primjeri:**

```
$ hostname
baltazar.srce.hr
$ dnsdomainname
srce.hr
```

Napomena

Brojni su razlozi zašto neki udaljeni sustav može biti nedostupan. Na primjer:

- greška (hardverska ili softverska) na lokalnom sustavu
- fizički prekid veze na lokalnoj mreži ili negdje drugdje na mrežnom putu do udaljenog sustava
- greška u DNS-poslužitelju (nemogućnost prevođenja simboličkog imena udaljenog stroja u numeričku adresu)
- administrativna zabrana na lokalnom ili udaljenom sustavu ili mreži
- greška na udaljenom sustavu
- isključen udaljeni sustav.

Naredbe opisane u ovom poglavlju mogu nam pomoći u otkrivanju uzroka problema, ako do njega dođe.

10.3. Prevođenje mrežnih adresa: nslookup, host i dig

Naredbama **nslookup**, **host** i **dig** upućujemo upite DNS-poslužiteljima te prevodimo simbolička imena sustava u numeričke adrese (i obrnuto).

Sintaksa:

```
nslookup [UPIT]
host [-t TIP] UPIT [DNS-POSLUŽITELJ]
dig [@DNS-POSLUŽITELJ] SIMBOLIČKO_IME [TIP]
dig [@DNS-POSLUŽITELJ] -x ADRESA
```

Na taj način možemo doznati podatke poput:

- numeričke adrese stroja na mreži:


```
nslookup baltazar.srce.hr
host jagor.srce.hr
dig www.srce.hr
```
- simboličkog imena stroja kojemu poznajemo numeričku adresu:


```
nslookup 161.53.2.82
host 161.53.2.130
dig -x 161.53.2.82
```
- podataka o nadležnom DNS-poslužitelju ili e-mail poslužitelju:


```
host -t NS srce.hr
host -t MX mzos.hr
dig srce.hr NS
```

Primjeri:

```
$ nslookup baltazar.srce.hr
Server:          161.53.8.10    ← podaci o DNS-poslužitelju
Address:         161.53.8.10#53 ← koji odgovara na
                                postavljenu upit

Non-authoritative answer:
Name:   baltazar.srce.hr      ← odgovor na postavljenu
Address: 161.53.2.82         upit

$ host -t NS srce.hr
srce.hr name server bjesomar.srce.hr
srce.hr name server regoc.srce.hr
srce.hr name server wcs.srce.hr.
```

↑
rezultat pokazuje da su za domenu srce.hr nadležna tri DNS-poslužitelja

```
$ dig marun.srce.hr

; <<>> DiG 9.8.1-P1 <<>> marun.srce.hr
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
52987
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3,
                        ADDITIONAL: 5

;; QUESTION SECTION:
marun.srce.hr.                IN      A

;; ANSWER SECTION:
marun.srce.hr.                14400   IN      A      161.53.2.76

;; AUTHORITY SECTION:
srce.hr.                      14400   IN      NS      regoc.srce.hr.
srce.hr.                      14400   IN      NS      bjesomar.srce.hr.
srce.hr.                      14400   IN      NS      wcs.srce.hr.

;; ADDITIONAL SECTION:
wcs.srce.hr.                  14400   IN      A      193.198.155.5
regoc.srce.hr.                14400   IN      A      161.53.2.69
regoc.srce.hr.                14400   IN      AAAA    2001:b68:c:2::69:0
bjesomar.srce.hr.             14400   IN      A      161.53.2.70
bjesomar.srce.hr.             14400   IN      AAAA    2001:b68:c:2::70:0

;; Query time: 1 msec
;; SERVER: 161.53.2.70#53(161.53.2.70)
;; WHEN: Wed Jan  2 14:32:46 2013
;; MSG SIZE rcvd: 212
```

10.4. Provjera dostupnosti udaljenih sustava: ping i traceroute

Naredbama **ping** i **traceroute** provjeravamo dostupnost udaljenih sustava na mreži.

Sintaksa:

```
ping [-c BROJ_PAKETA] DESTINACIJA
traceroute DESTINACIJA
```

Naredba **ping** šalje posebne mrežne pakete (ICMP) na zadanu destinaciju, čeka odgovor od destinacije i ispisuje vrijeme potrebno za primitak odgovora. Ukoliko se navede broj paketa, naredba **ping** će slati pakete sve dok se ne prekine njeno izvođenje ([Ctrl-C]). Na kraju naredba **ping** ispisuje jednostavne statističke podatke vezane uz odašiljanje i primanje paketa.

Naredba **traceroute** ispisuje mrežnu stazu do zadane destinacije.

Primjeri:

```
$ ping -c 2 regoc.srce.hr
PING regoc.srce.hr (161.53.2.69): 56 data bytes
64 bytes from 161.53.2.69: icmp_seq=0 ttl=64 time=0.1 ms
64 bytes from 161.53.2.69: icmp_seq=1 ttl=64 time=0.1 ms

--- regoc.srce.hr ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.1/0.1/0.1 ms

$ traceroute fly.srk.fer.hr
traceroute to fly.srk.fer.hr (161.53.74.66), 30 hops
max, 38 byte packets
 1 c7000b (161.53.2.65)  0.264 ms  0.190 ms  0.179 ms
 2 193.198.229.70 (193.198.229.70)  7.598 ms  7.235 ms  8.055 ms
 3 193.198.229.10 (193.198.229.10)  0.493 ms  0.285 ms  0.271 ms
 4 161.53.16.14 (161.53.16.14)  5.578 ms  3.740 ms  7.712 ms
 5 fly.srk.fer.hr (161.53.74.66)  0.576 ms  0.390 ms  0.369 ms
$
```

10.5. Vježba: *Linux* na mreži

1. Koje je mrežno ime sustava na kojem redite? U kojoj se domeni on nalazi?
2. Pokrenite naredbu `uname -a`. Što možete reći o sustavu na kojem radite?
3. Pokrenite naredbu `nslookup` i potražite odgovore na sljedeća pitanja:
 - Koja je numerička adresa sustava na kojem trenutno radite?
 - Kojim se DNS-poslužiteljem služi sustav na kojem trenutno radite?
 - Što možete reći o adresi `www.srce.hr`?
4. Pokrenite naredbu `host www.google.com`. Što možete reći o adresi `www.google.com`?
5. Koliko je DNS-poslužitelja nadležno za hrvatsku vršnu domenu (`.hr`)?
6. Provjerite dostupnost stroja `regoc.srce.hr` i njegovu brzinu odziva.
7. Pomoću programa `ssh` prijavite se na sustav pod istim korisničkim imenom.
 Proučite koje podatke o Vama ispisuju naredbe `who` i `finger`. Koliko ste puta prijavljeni na sustav? Možete li u popisima koje su ispisale naredbe `who` i `finger` razlikovati svoje različite prijave?
8. Upišite `exit` i odjavite se sa sustava (napravite to samo jednom).
 Nakon što se izvršila naredba `exit`, jeste li i dalje prijavljeni na sustav?

10.6. Pitanja za ponavljanje

1. Čemu služe naredbe `hostname` i `dnsdomainname`?
2. Koje podatke ispisuje naredba `uname -a`?
3. Čemu služi naredba `nslookup`?
4. Koja je razlika između naredbi `ping` i `traceroute`?
5. Iz kojih razloga neki sustav može biti nedostupan putem mreže?
6. Koja je razlika između naredbi `ssh` i `scp`?

A. Dodatak: Mrežne postavke sustava *Linux*

Napomena

Kućne lokalne mreže najčešće omogućavaju automatsko dodjeljivanje mrežnih adresa i drugih mrežnih postavki. To omogućava protokol koji se zove DHCP.

Postupak instalacije većine sustava *Linux* podržava ovakve mreže te u tom slučaju nakon instalacije sustava obično nije potrebno raditi nikakve izmjene na sustavu.

Ako je izmjene ipak potrebno raditi, tada se one obično rade pomoću grafičkih alata za administraciju sustava.

A.1. Preduvjeti za rad na mreži

Računala ostvaruju vezu s mrežom i drugim računalima putem mrežnih sučelja. Sustavi *Linux* mogu imati jedno ili više mrežnih sučelja koja se standardno označavaju s `eth0`, `eth1` i tako dalje.

Da bi se računalo uspješno spojilo na mrežu, potrebno je:

- mrežnom sučelju dodijeliti mrežnu adresu i mrežnu masku
- zadati izlazni usmjerivač
- zadati DNS-poslužitelj putem kojeg će se prevoditi simboličke u brojčane mrežne adrese.

Način postavljanja mrežnih postavki ovisi o inačici sustava koji se koristi.

Za postavljanje i izmjenu mrežnih postavki potrebno je imati administratorske ovlasti.

A.2. Mrežna adresa računala i izlazni usmjerivač (*gateway*)

Naredbom `ifconfig` mogu se mijenjati postavke mrežnih sučelja.

Ako se naredba `ifconfig` navede bez dodatnih opcija, ispisat će popis svih mrežnih sučelja i njihove postavke.

Sintaksa:

```
ifconfig
ifconfig SUČELJE [ADRESA [netmask MASKA]] [up|down]
```

Primjeri:

ispis postavki sučelja eth0

↓

```
# /sbin/ifconfig eth0
eth0 Link encap:Ethernet HWaddr 08:00:27:BA:BC:1B
      inet addr:10.0.2.15 Bcast:10.0.2.255 Mask:255.255.255.0
      inet6 addr: fe80::a00:27ff:feba:bc1b/64 Scope:Link
      UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
      RX packets:115 errors:0 dropped:0 overruns:0 frame:0
      TX packets:96 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:27895 (27.2 KiB) TX bytes:9786 (9.5 KiB)

# ifconfig eth0 10.0.2.15 netmask 255.255.255.0
```

↑

postavljanje adrese 10.0.2.15 s mrežnom maskom 255.255.255.0 na sučelje eth0 - navedene će postavke vrijediti samo do sljedećeg pokretanja sustava

Naredbom **route** može se zadati izlazni usmjerivač (*gateway*) putem kojeg će se ostvariti veza s Internetom. Ako se naredba zada bez argumenata, ispisat će se popis poznatih mrežnih ruta.

Sintaksa:

```
route
route add -net default gw ADRESA
```

Primjeri:

```
# route add -net default gw 10.0.2.2

# route
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
10.0.2.0 * 255.255.255.0 U 1 0 0 eth0
default 10.0.2.2 0.0.0.0 UG 0 0 0 eth0
```

↑
adresa izlaznog usmjerivača

default *označava da se radi o izlaznom usmjerivaču*

Navedene se postavke trajno postavljaju zapisima u odgovarajućim konfiguracijskim datotekama sustava. Imena datoteka i sintaksa ovise o inačicama sustava *Linux*.

U slučaju sustava **Red Hat**, **Fedora** i **CentOS** podatak o mrežnoj adresi upisuje se u `/etc/sysconfig/network-scripts/ifcfg-eth0` (za mrežno sučelje `eth0`). Na primjer:

```
DEVICE=eth0
BOOTPROTO=static
IPADDR=10.0.2.15
NETMASK=255.255.255.0
ONBOOT=yes
```

Podatak o osnovnom izlaznom usmjerivaču upisuje se u datoteku `/etc/sysconfig/network`. Na primjer:

```
GATEWAY=10.0.2.2
```

Nakon toga potrebno je zadati `/etc/init.d/network restart` ili ponovno pokrenuti sustav.

U slučaju sustava **Debian** i **Ubuntu** isti se podaci upisuju u samo jednu datoteku (`/etc/network/interfaces`). Na primjer:

```
iface eth0 inet static
    address 10.0.2.15
    network 10.0.2.0
    netmask 255.255.255.0
    broadcast 10.0.2.255
    gateway 10.0.2.2
```

A.3 Simboličko ime računala i DNS-poslužitelji

Simboličko ime računala na sustavima *Red Hat*, *Fedora* i *CentOS* upisuje se u datoteku `/etc/sysconfig/network`:

```
HOSTNAME=linux.abc.hr
```

a na sustavima *Debian* i *Ubuntu* u datoteku `/etc/hostname`:

```
linux.abc.hr
```

Podaci o DNS-poslužitelju zapisuju se u datoteku `/etc/resolv.conf`.

Na primjer:

```
search abc.hr  
nameserver 161.53.123.2
```

U prvom je retku zadano da se prilikom prevođenja simboličkih imena u brojčane adrese svim adresama koje ne sadrže točku doda nastavak `abc.hr` (na primjer `www` će biti pretvoreno u `www.abc.hr`), a u drugom je retku zadana adresa DNS-poslužitelja.

