ZAVOD ZA ELEKTRONIKU, MIKROELEKTRONIKU, RAČUNALNE I INTELIGENTNE SUSTAVE
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA
SVEUČILIŠTE U ZAGREBU

G N U / L I N U X

Osnovni tečaj

uredili: Marko Salkić

Miljen Mikić

Zagreb, 2005.

Sadržaj

Zahvaljujemo se svima koji su pomogli izradi ove zbirke, a posebno kolegi Dinku Koruniću koji je uvelike pomogao svojim iskustvom i savjetima, mr.sc. Stjepanu Grošu kao inicijatoru, koji nas je vodio i savjetovao tokom izrade cijelog projekta, Ani Kukec, Luki Drvoderiću, Božidaru Jurici Kenigu, te svima ostalima koji su sudjelovali na izradi ovog projekta.

Linux zemris team

1. Osnove Linuxa

1.1. Operacijski sustav

Operacijski sustav (*operating system*) je sustav programske podrške (*software*) na računalu, koji upravlja radom sklopovlja (*hardware*), te čini vezu između korisnika (*user*) i sklopovlja. Nadalje, operacijski sustav čini platformu (bazu), na kojoj se drugi programi, zvani **aplikacije**, mogu izvoditi. U širem smislu operacijski sustav se sastoji od 3 dijela: korisničkog sučelja, sistemskih alata i jezgre.

1.2. Linux

Linux je operacijski sustav koji omogućava korisnicima rad s aplikacijama, sklopovskim resursima, i komunikaciju sa ostalim računalnim sistemima. Više korisnika može istovremeno raditi na Linuxu, što znači da je on **višekorisnički** (*multiuser*) operacijski sustav. Nadalje, svaki od tih korisnika može pokretati više aplikacija odjednom jer je Linux **višezadaćni** (*multitasking*) operacijski sustav.

Linux je slobodna implementacija POSIX (**P**ortable **O**perating **S**ystem **I**nterface, a **X** stoji od Unix) specifikacije operativnog sustava, napisana od nule (tj. ne sadrži kod od UNIX-a). Slobodno je dostupan i njegov je izvorni kod pristupačan svakome. To omogućava korisnicima da ga maksimalno prilagode svojim potrebama.

Jedno od glavnih načela Linuxa je, uz neke iznimke, da on sve resurse sa kojima raspolaže predstavlja kao datoteke. On omogućava vrlo jednostavno preusmjeravanje ulaza/izlaza i ulančavanje (ref. 7. pogl). Tako na primjer Linux predstavlja tvrdi disk kao jednu datoteku u koju se može pisati ili iz nje čitati, monitor kao drugu, a pisač kao treću datoteku. Sve te datoteke su u novije vrijeme, zbog porasta samog operacijskog sistema, hijerarhijski organizirane u obliku stabla.

1.3. Struktura Linuxa

U širem smislu pod terminom Linux se obično podrazumijeva višekorisnički, višezadaćni operacijski sustav koji se sastoji od različitih programskih cjelina kao što su npr. editori, programski prevodioci, različiti namjenski programi. Tada govorimo o Linux distribuciji. U užem smislu Linux je samo jezgra operativnog sistema koji upravlja resursima računala. Osnovna organizacijska struktura Linuxa dana je slikom 1.1

Slika 1.1: Osnovna organizacijska struktura

1.3.1. Jezgra (*kernel*) Linuxa

Kao i svakog drugog operativnog sustava, jezgra je dio čiji je osnovni zadatak kontrola sklopovskog dijela računala (upravljanje radom procesora, dodjela memorije i sl.). Kad pričamo o jezgri, valja napomenuti da Linux koristi dva načina rada. To su nadzorni i korisnički način rada. Oni postoje zato da aplikacije koje se pokreću iz korisničkog načina ne bi oštetile operacijski sustav. Ta kontrola je izvedena i hardverski, u procesoru. U nadzornom načinu se može direktno upravljati sa npr. memorijom, IRQ, DMA. On ujedno služi da bi se postavila ograničenja nad aplikacijama koje se pokreću iz korisničkog načina. Korisnički način rada je zamišljen kao radna okolina.

Jezgra operacijskog sustava može biti građena na dva načina: mikrojezgra i monolitna jezgra. Mikrojezgra je građena tako da se najnužniji softver za pokretanje operacijskog sustava nalazi u njoj. Takva jezgra ne pruža mnogo funkcionalnosti, samo onu osnovnu. Zato uvodimo pojam **modula**. Modul je program koji se prilikom pokretanja Linuxa smješta u radnu memoriju, i kojeg koristi jezgra da omogući određenu funkcionalnost. Na primjer, kada se operacijski sustav pokrene, on ne podržava umrežavanje, no ukoliko se modul koji to podržava pokrene, umrežavanje biva podržano operacijskim sustavom. Moduli koji se izvode nemaju sva prava koja ima jezgra. Oni se izvode iz korisničkog načina rada, ali imaju više prava od korisničkih aplikacija. Prednost ovakvog pristupa je taj da kada dođe do greške u radu nekog modula, on u principu ne utječe na rad operacijskog sustava. Još jedna prednost je lakše održavanje. Mana ovakve izvedbe jezgre je ta da ima slabije performanse od monolitne jezgre.

Monolitna jezgra je izvedba jezgre u kojoj su sve funkcionalnosti ugrađene u nju. Ona radi kao jedna cjelina. Za dodavanje, uklanjanje ili promjenu neke funcionalnosti na razini operacijskog sustava, potrebno je cijelu jezgru izgraditi ponovo (rekompajlirati). Glavna razlika od mikrojezgre je ta da u ovoj izvedbi jezgre svi moduli koji su ugrađeni u nju imaju sva prava nadzornog načina rada. Bitna prednost je ta što ima vrlo visoke performanse u odnosu na mikrojezgru. Jezgra Linuxa je monolitska jezgra.

1.3.2. Ljuska (*shell*) i korisnička razina

Su dio sustava gdje se nalaze aplikacije kojima se služi korisnik za razvoj svojih programa, rad sa

datotekama, komunikaciju, dakle sve što Linux omogućuje. Ljuska je program koji služi kao tumač (interpreter) naredbi i služi kao veza između korisnika i Linuxa. Ljuska je u stvari programski jezik sa varijablama, kontrolnim naredbama, potprogramima, prekidima i sl. Svaki korisnik ima odvojenu kopiju ljuske pa prema tome ljuske mogu biti različite za svakog korisnika. Neke ljuske su na primjer: Bourne shell, razvijana u okviru firme AT&T. Poboljšane verzije te ljuske su Korn i C ili BSD ljuska koja je razvijena na University of California at Berkley. Danas su najpopularnije Bash i Zsh.

1.4. Kratki povijesni razvoj Linuxa

Vodeći se idejom o stvaranju slobodnog softvera koji će se moći slobodno distribuirati i čiji će izvorni kod biti dostupan korisnicima sofvera, Richard Stallman je 1983. osnovao GNU Projekt. GNU projekt je imao za cilj stvaranje slobodnog operativng sustava kompatibilnog s Unixom. Do 1990. su u sklopu GNU projekta dovršeni svi potrebni dijelovi sustava osim jezgre. Jezgra na kojoj su sami radili, GNU Hurd kernel, je napredovala sporo i nije zadovoljavala. U međuvremenu je Linus Torvalds radio na Linux kernelu, također slobodnom softveru koji je poslužio GNU Projektu. Cjelovito ime Linuxa u širem smislu je zato GNU/Linux. Linux kernel je djelo studenta računarstva Linusa Torvaldsa, nastao 1991. godine kao posljedica Linusovog hobija. Linus je tada imao tek 23 godine. Nadao se da će stvoriti robusnu inačicu UNIX-a za Minix korisnike.

Sustav Minix predstavlja program kojeg je razvio profesor računarskih znanosti Andrew Tannebaum. Sustav Minix je napisan u svrhu učenja, a da bi prikazivao nekoliko koncepata vezanih uz operacijske sustave. Torvalds je te koncepte objedinio u samostalan sustav nalik UNIX-u. Do tog sustava su mogli doći svi širom svijeta.

Linus Torvalds se potrudio ponuditi svojim kolegama korisnicima Minixa bolju platformu koju je bilo moguće pokretati na osobnim računalima. Linus se opredijelio za računala s procesorom 386 jer je taj procesor imao na raspolaganju.

Prije pojave prvih distribucija, korisnik Linuxa je morao znati koji se sve dijelovi operacijskog sustava moraju pokrenuti da bi se sustav pokrenuo. To je bilo jako nepraktično. Čim se upotreba Linuxa počela širiti, pojavile su se i prve distribucije (operacijski sustav nalik Unix-u koji uključuje Linux jezgru, i niz popratnih programa). Distribucija Linuxa je zamišljena da olakša korisnicima korištenje Linuxa. Trenutno u razvoju postoji više od 200 distribucija.

Prve distribucije poput MCC Interim Linux (razvijena na University of Manchester, 1992.), TAMU (Texas A&M University, razvijena u isto vrijeme) i SLS (Soflanding Linux system) su bile teške za održavanje. Distribucija Slackware autora Patricka Volkerding, baziranu na SLS distribuciji, je nastarija distribucija koja se još nadograđuje.

Neke danas najraširenije i najupotrebljavanije Linux distribucije su [6,7]:

- CentOS 4.2
- Debian 3.1
- Fedora 4
- Gentoo 2005.

1.5. Tko je vlasnik Linuxa?

IBM posjeduje OS/2, a Microsoft MS-DOS i MS Windowse, ali tko je vlasnik Linuxa? Prvo i osnovno, razni ljudi polažu pravo na različite komponente Linuxa. Tvrtka Red Hat je vlasnik Red Hat distribucije, a Patrick Volkerding distribucije Slackware. Mnogi Linux uslužni programi spadaju pod licencu o javnom korištenju General Public License (GPL) kako se on i distribuira. U osnovi, Linus i većina tvoraca Linuxa su svoje djelo zaštitili pomoću te licence koja se može pronaći i na Internetu. Ta se licenca ponekad može naći i pod imenom GNU Copyleft (pandan riječi *copyright*). Njome je programerima omogućeno stvaranje softvera za bilo koga i osnovna joj je pretpostavka da bi softver trebao pripadati svima tako da bilo tko može prepraviti program prema svojim potrebama. Jedini je uvjet da i ostali korisnici imaju prava na taj novi, promijenjeni programski kôd. GNU Copyleft, ili GPL, olakšava programerima rad, upravo zbog principa otvorenog koda, ali i korisnicima mijenjanje i prodavanje novih programa. Izvorni programeri ne mogu ograničiti ista takva prava na mijenjanje programa ljudima koji kupuju program. Ukoliko prodate program u istom ili izmijenjenom obliku, morate isporučiti i izvorni kôd programa. Zbog toga Linux uvijek možete dobiti zajedno s njegovim izvornim kôdom.

1.6. Korištenje sustava pomoći

1.6.1. Naredba `man`

Man (skraćenica od eng. manual) je program koji daje korisniku informacije iz dostupnog priručnika. Njime dobivamo detaljnije informacije o tome kako se zadaje pojedina naredba, što ona radi, kakve dodatne parametre (opcije) ima dotična naredba i sl. Sintaksa naredbe man je sljedeća:

```
man -k <ključne_riječi>
man -f <ime_datoteke>
man <ime_naredbe>
```

U slučaju prve varijante sintakse, na ekranu će biti ispisan kratak opis svake naredbe koja bilo gdje u opisu ima neku od ključnih riječi. U drugom slučaju program će potražiti opis vezan uz zadanu datoteku, dok će se u trećem ispisati opis navedene naredbe. Naredba man može primiti još mnogo parametara, no oni neće biti sagledani (za detaljnije informacije pogledati na literaturu iz popisa korištene literature).

Priručnik se u Linux-u sastoji od nekoliko cjelina, koje se mogu specificirati navođenjem broja cjeline iza ključne riječi `man`. Cjeline su prikazane u tablici 1.1.

Tablica 1.1: Cjeline priručnika man

Broj cjeline	Kratak opis
1	Naredbe dostupne korisnicima
2	Linux i C sistemske naredbe
3	Funkcije za C programe
4	Specijalni nazivi datoteka
5	Tipovi datoteka i konvencije vezane uz Linux
6	Igre
7	Paketi vezani uz procesiranje teksta
8	Administracijske naredbe i procedure

Primjer:

```
$ man ls
LS(1)                          User Commands
LS(1)
NAME
       ls - list directory contents
SYNOPSIS
       ls [OPTION]... [FILE]...
DESCRIPTION
       List  information  about  the FILEs (the current directory by
default).
       Sort entries alphabetically if none of -cftuSUX nor –sort.
       Mandatory arguments to long options are  mandatory  for  short
options
       too.
...
```

Primjer:

```
$ man -f ls
ls (1)               - list directory contents
```

Primjer:

```
$ man -k ls
blockdev (8)         - call block device ioctls from the command line
charmap (5)          - character symbols to define character encodings
deallocvt (1)        - deallocate unused virtual terminals
dircolors (1)        - color setup for ls
false (1)            - do nothing, unsuccessfully
fdutilsconfig (8)    - configure the suid bit of fdmount
get_kernel_syms (2)  - retrieve exported kernel and module symbols
intro (2)            - Introduction to system calls
kallsyms (8)         - Extract all kernel symbols for debugging
ksyms (8)            - display exported kernel symbols.
logrotate (8)        - rotates, compresses, and mails system logs
ls (1)               - list directory contents
lsattr (1)           - list file attributes on a Linux second extended
file system
lsmod (8)            - list loaded modules.
lspci (8)            - list all PCI devices
man (1)              - an interface to the on-line reference manuals
md5sum.textutils (1) - compute and check MD5 message digest
protocols (5)        - the protocols definition file
shells (5)           - pathnames of valid login shells
signal (7)           - list of available signals
textdomain (3)       - set domain for future gettext() calls
tty_ioctl (4)        - ioctls for terminals and serial lines
watch (1)            - execute a program periodically, showing output
fullscreen
```

Primjer:

```
$ man 2 mount
MOUNT(2)                              Linux Programmer's Manual
MOUNT(2)
NAME
       mount, umount - mount and unmount filesystems
SYNOPSIS
       #include <sys/mount.h>
       int  mount(const  char *source, const char *target, const char
*filesystemtype, unsigned long
       mountflags, const void *data);
       int umount(const char *target);
       int umount2(const char *target, int flags);
DESCRIPTION
       mount attaches the filesystem specified by source (which is often
a device name, but can
also
       be a directory name or a dummy) to the directory specified by
target.
       umount and umount2 remove the attachment of the (topmost)
filesystem mounted on target.

...
```

Primjer:

```
$ man 8 mount
MOUNT(8)                  Linux Programmer's Manual
MOUNT(8)
NAME
       mount - mount a file system
SYNOPSIS
       mount [-lhV]
       mount -a [-fFnrsvw] [-t vfstype] [-O optlist]
       mount [-fnrsvw] [-o options [,...]] device | dir
       mount [-fnrsvw] [-t vfstype] [-o options] device dir
DESCRIPTION
       All files accessible in a Unix system are arranged in one big
tree, the
       file hierarchy, rooted at /.  These files can be spread out  over
sev-
       eral  devices. The mount command serves to attach the file system
found
       on some device to the big file tree. Conversely, the umount(8)
command
       will detach it again.
       The standard form of the mount command, is
              mount -t type device dir
...
```

1.6.2. Naredba `info`

Info je program razvijen unutar GNU projekta za čitanje datoteka tipa Info. Prvo se pišu datoteke u Textinfo jeziku, koje se zatim prevode u Info tip datoteka. Program info datoteke tipa Info interpretira u čitljiv tekst (slično kao HTML). Sintaksa naredbe info je:

```
info [<opcije>] [klju ne_rije i]
```

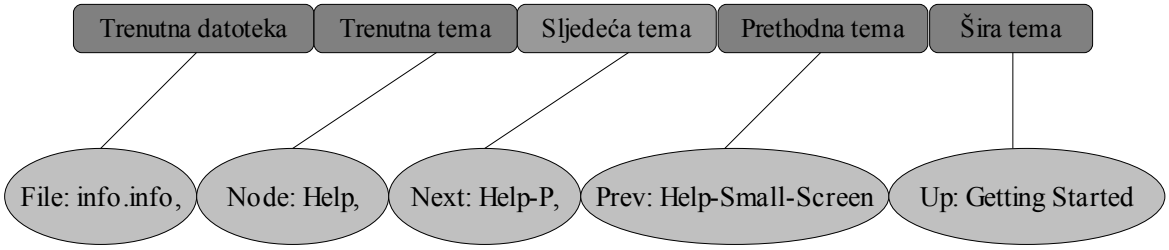
Ovisno o navedenoj opciji i ključna riječ ima drugačije značenje. Ukoliko se ne navede nikakva opcija, a navede ključna riječ, program info će pretražiti direktorije navedene u INFOPATH, i ako se odgovarajuća datoteka tamo nalazi, program će prikazati dostupne informacije o traženom pojmu. Ako ne pronađe odgovarajuću datoteku, program će se pokrenuti kao da niti jedna ključna riječ nije navedena. Korisnik može dodati direktorij u INFOPATH naredbom:

```
info -d <direktorij>
```

Ili pak, ukoliko samo želi pogledati datoteku, za koju zna da nije u direktoriju koji info pretražuje, to može naredbom:

```
info -f <ime_datoteke>
```

Vrlo je važno znati se snalaziti u programu info. Navigacija programom odvija se tipkama kao što je prikazano na slici 1.1):



Slika 1.2. Zaglavlje programa info

Navigacija je omogućena tipkama prikazanim u tablici 1.2.

Tablica 1.2: Tipke naredbe info

Tipke	Značenje
n	prikaz sljedeće teme u dokumentaciji
p	prikaz prethodne teme u dokumentaciji
<SPC>	prikazuje sljedeću stranicu dokumentacije (ako cijela tema ne stane na ekran)
<BAKSPACE>	prikazuje prethodnu stranicu dokumentacije
u	prikazuje hijerarhijski višu temu
q	izlazi iz programa

2. Korisnici u Linux operacijskom sustavu

2.1. Terminal

Općenito, terminal je uređaj koji korisniku omogućuje komunikaciju s računalnim sustavom. U ovom tekstu pojam terminal ponajprije će se koristiti u smislu programa koji služi kao posrednik između korisnika i poslužitelja. Drugim riječima, terminal će omogućiti nekom lokalnom ili udaljenom (eng. remote) korisniku pristup glavnom Linux poslužitelju (eng. server).

Emulator terminala je konkretno program koji emulira ("glumi") terminal unutar neke druge sistemske arhitekture. Tako npr. popularni klijent PuTTY pod Windows operacijskim sustavom služi za udaljenu emulaciju terminala prilikom prijave na neki poslužitelj. Primjeri emulatora terminala za Linux su Xterm, rxvt, gnome-multi-terminal, gnome-terminal, kconsole, aterm, itd.

Postoji više vrsta terminala. Osnovna podjela je na tekstualne, koji mogu prikazivati samo tekst, i na grafičke, koji osim teksta mogu prikazivati i grafiku. Vjerojatno najrašireniji terminal je popularni DEC-ov VT100 iz 80-ih godina prošlog stoljeća koji je praktički postao standardom za emulatore terminala.

2.2. Prijava i odjava sa sustava

Linux je višekorisnički operacijski sustav što znači da ga više korisnika može koristiti istovremeno. Svaki korisnik ima svoje korisničko ime kako bi ga sustav mogao identificirati i zaporku kao sigurnosni element koji štiti njegove podatke. Uspješna prijava u sustav podrazumijeva pravilan unos kombinacije korisničkog imena i zaporke što omogućuje daljnji rad na sustavu. Odjava sa sustava u načelu "zaključava" korisnikove podatke do sljedeće prijave.

Treba naglasiti jednoznačno dodjeljivanje korisničkih imena pojedinih korisnicima. Također, treba istaknuti da je Linux *case sensitive* tj. da je osjetljiv na velika i mala slova ("Marko" i "marko" nije isto). Najčešće su korisnička imena pisana malim slovima te su kombinacija imena i prezimena (ili inicijala) nekog korisnika. Na FER-ovom serveru `pinus.cc.fer.hr` korisničko ime je kombinacija inicijala i JMBAG-a tog korisnika. Na ovaj se način uklanja mogućnost pojavljivanja dvaju istih korisničkih imena.

2.2.1. Naredba `login`

Naredba `login` služi za prijavu na Linux operacijski sustav. Iako je uobičajeno da se nakon uključivanja računala i podizanja sustava automatski otvara tekstualno ili grafičko sučelje u kojem korisnik jednostavno upisuje svoje korisničko ime i zaporku, smisao ove naredbe leži i u tome da sustav nakon podizanja omogućava jednostavnu promjenu trenutnog korisnika.

2.2.2. Naredba `logout`

Ova naredba služi za odjavljivanje sa sustava. Jednostavno se upiše `logout` i sustav nas odjavljuje ili vraća u sučelje (ljusku) korisnika sa čijeg smo terminala upisivali naredbu `login`.

2.3. Promjena zaporke

Zaporka (eng. password) ili samo njezino mijenjanje jedna je od važnijih stvari oko kojih bi običan korisnik trebao voditi računa prilikom korištenja Linux sustava. Zaporka je riječ sastavljena od slova i znakova koja omogućava pojedinom korisniku da uopće koristi sustav. Bilo bi dobro da se zaporka sastoji od malih i velikih slova, brojeva i dodatnih znakova jer se tako smanjuje mogućnost da dođe u

krive ruke. Posljedice koje mogu nastati ako zaporka dođe u ruke zlonamjernih osoba mogu biti katastrofalne ne samo za korisnika već i za cijeli sustav. Dobra preporuka je da se odabere zaporka koja nema nikakve povezanosti s korisnikom ni sa događajima i stvarima iz njegovog života. Evo primjera: neka korisnik Ivica ima doma psa imena Fido. Nikako nije dobro da njegova zaporka za prijavu na sustav bude "Fido". Ako se već ide na simboliku, tada bi zaporka "F3ed0" bila puno bolja. Ovaj primjer izgleda pomalo smiješno, ali postoje i banalnije stvari preko kojih se dolazi do zaporke. Još jedan koristan savjet je taj da zaporku ne ostavljate negdje fizički zapisanu već da ju pamtite ukoliko je to moguće.

2.3.1. Naredba `passwd`

Naredba `passwd` služi za promjenu zaporke na sustavu. Promijeniti zaporku korisnik može tek nakon što se prijavi na sustav. Promjena zaporke vrši se tako da se u sučelju upiše naredba `passwd` nakon čega sustav traži da upišete trenutnu važeću zaporku, a zatim novu zaporku. Iz sigurnosnih razloga (zbog krivog unosa nove zaporke) sustav će zatražiti da novu zaporku upišemo još jednom. Nakon ovoga, naša se zaporka promijenila te ćemo morati koristiti novu zaporku prilikom narednih prijava na sustav.

```
$ passwd
Changing password for user luka.
Changing password for luka
(current) UNIX password:
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

2.4. Informacije o korisniku

2.4.1. Naredba `finger`

Uobičajeno je da među skupinom ljudi koji su na bilo kakav način povezani u nekakvu grupu postoji određena interakcija. Interakcija unutar određene grupe ljudi ostvariva je uz uvjet da svaka osoba iz grupe posjeduje elementarne informacije o svakom članu grupe. Linux je višekorisnički sustav pa shodno tome omogućava pohranu osnovnih podataka o korisnicima. Ovi su osnovni podaci dovoljni za komunikaciju među korisnicima sustava. Unose se prilikom ostvarivanja novog korisničkog računa na sustavu, a najčešće sadrže informacije poput imena i prezimena, adrese, broja telefona i sličnog.

Informacije o određenom korisniku mogu se dobiti ako se u terminalu upiše:

```
finger <korisničko ime>
```

Sustav nam kao rezultat izlistava podatke o traženom korisniku. Osim imena i prezimena, to su i podaci o terminalu koji korisnik upotrebljava, vrijeme kad je traženi korisnik zadnji put bio prijavljen na sustav te adresu s koje se prijavio. Ako upišemo samo `finger` dobit ćemo listu imena i prezimena svih korisnika trenutno prijavljenih na sustav.

```
$ finger
Login   Name            TTY   Idle    When    Where
tb      Tomislav Biscan pts/2    8 Mon 18:46  dh75-63.xnet.hr
tb      Tomislav Biscan pts/4   18 Mon 18:46  gnjilux.srk.fer.hr
wgottwe Walter Gottwein pts/8    4 Mon 18:50  gnjilux.srk.fer.hr
llovos  Luka Lovosevic  pts/14     Mon 21:55  ad16-m135.net.t-com.hr
```

Drugi primjer prikazuje detaljne podatke o korisniku `llovos`: korisničko ime, ime i prezime, matični direktorij, ljusku koju koristi, datum, vrijeme, terminal i adresu zadnje prijave na sustav, vrijeme koliko je neaktivan te ima li nepročitanu elektroničku poštu.

```
$ finger llovos
Login name: llovos                  In real life: Luka Lovosevic
Directory: /home/users/l/llovos         Shell: /bin/sh
On since May 23 21:55:42 on pts/14 from ad16-m135.net.t-com.hr
20 seconds Idle Time
No unread mail
No Plan.
```

2.4.2. Naredba `chfn`

Ovom naredbom možemo promijeniti osvnovne informacije o pojedinim korisnicima. Te su informacije ime i prezime, ured, broj telefona u uredu i kućni telefonski broj. Ukoliko se iza imena naredbe ne navede parametar, mijenjaju se informacije korisnika koji je pokrenuo naredbu.

```
$ chfn
Changing finger information for luka.
Password:
Name [Luka Lovosevic]: Drugo Ime
Office [Doma]: Kuca
Office Phone []: 0800600600
Home Phone [ble]: 014683377

Finger information changed.
```

2.4.3. Naredba `who am i`

Ukoliko iz bilo kojeg razloga želite saznati tko je trenutni korisnik sustava odnosno terminala, to možete učiniti na više načina. Mogućnost odabira između više načina rješavanja danog problema je još jedna posebnost Linuxa. Jedan od načina je korištenje naredbe `who` uz argumente `am i`, dakle `who am i`. Rezultat izvođenja je prikazan u više polja. Prvi dio je korisničko ime, drugi je trenutna komunikacijska linija, tj. terminal koji se koristi, treći je datum i vrijeme prijave u sustav i zadnji je adresa s koje se pristupa sustavu. Isti efekt daje i naredba `who -m`. Osim ovih, postoji i naredba `whoami` (bez razmaka) koja vraća korisničko ime bez ostalih detalja.

```
$ who am i
luka      pts/1      May 23 23:30
$ whoami
luka
```

2.5. Informacije o ostalim korisnicima

Često će u isto vrijeme više korisnika biti prijavljeno na sustav. Ponekad je korisno znati tko su ti korisnici, te kada su se prijavili, na koji terminal i slično.

2.5.1. Naredba `users`

Ova je naredba pandan naredbe `whoami` – rezultat su razmakom odvojena korisnička imena koja su trenutno prijavljena na sustav bez ikakvih detalja.

```
$ users
luka
```

2.5.2. Naredba `who`

Prije spomenuta naredba `who` daje više detalja o trenutnim korisnicima. Osim korisničkog imena, saznajemo i terminal koji korisnik upotrebljava, datum i vrijeme prijave na sustav te adresu s koje se prijava izvršila. Puno više dodatnih informacija daje nam `who -a`, primjerice vrijeme zadnjeg podizanja sustava, vrijeme zadnje promjene sistemskog vremena, itd.

```
$ who
luka       :0           May 23 23:20
luka       pts/0        May 23 23:21
luka       pts/1        May 23 23:30
```

2.6. Aktivnost korisnika na sustavu

Da bi saznali što pojedini korisnici rade u danom trenutku, tj. koje procese imaju pokrenute, koristimo se naredbom `w` koja daje općenite informacije o sustavu: od kada je aktivan, koliko ima prijavljenih korisnika i slično. Rezultat izvođenja naredbe `w` podijeljen je po poljima. Ono što je ovdje novo u odnosu na `who` je stupac IDLE koji nam za pojedinog korisnika govori koliko je vremena prošlo u minutama i sekundama od zadnjeg pokretanja nekog programa. Sljedeća polja su JCPU i PCPU koja predstavljaju ukupni i trenutni procesorski utrošak korisnika izražen u minutama i sekundama (sufiksi `m` i `s`). Ono što nas je prvotno zanimalo nalazi se u stupcu WHAT i sadrži ime procesa kojeg korisnik ima trenutno pokrenutog. Ukoliko je proces pokrenut automatski prilikom prijave u sustav (poput ljuske), tada ime procesa sadrži prefiks '-'.

```
$ w
23:47:36 up 28 min,  3 users,  load average: 0,08, 0,06, 0,02
USER       TTY         LOGIN@      IDLE JCPU  PCPU WHAT
luka       pts/0       23:21       6:27 0.00s 0.54s kdeinit: kwrited
luka       pts/1       23:30       0.00s      0.14s       0.02s w
```

2.7. Promjena grupe

Struktura Linuxa je takva da svako korisničko ime pripada jednoj ili više grupa korisnika s time da ima jednu primarnu (glavnu) grupu dok su ostale sekundarne. Važnost grupa je u organiziranju svih korisnika sustava. Ovisno o tome kojoj grupi neki korisnik pripada, biti će mu dozvoljeno ili zabranjeno čitanje, pisanje ili pokretanje određenih datoteka na tom sustavu. Može se, međutim, javiti situacija kada korisnik želi promijeniti grupu kojoj trenutno pripada kako bi dobio dozvole koje mu omogućavaju neku radnju. Uvjet je, naravno, da pripada i grupi na koju se želi prebaciti. Ovo se izvodi naredbom `newgrp`. Argument je ime nove grupe, a ukoliko se takvo ime ne navede, naredba pretpostavlja primarnu (podrazumijevanu) grupu. Ukoliko korisnik ne pripada grupi na koju se želi prebaciti, biti će mu zabranjen pristup toj grupi (traži se upis zaporke za novu grupu).

```
$ newgrp root
Password:
Sorry.
```

2.8. Identifikator korisnika i grupe

Na razini sustava, za predstavljanje korisničkih imena i grupa koriste se brojevi. ID (eng. identification) je broj koji će služiti kao identifikacija korisnika u sustavu. Postoji UID (eng. user id) i GID (eng. group id). Prvi predstavlja broj koji jednoznačno određuje nekog korisnika. Npr. UID korisnika “marko” može biti 501, dok je UID korisnika “root” 0 (ovo je administrator sustava). Drugi broj, GID, je brojevni ekvivalent imena grupe kojoj korisnik pripada.

2.8.1. Naredba `id`

Ukoliko želimo vidjeti koji nam je UID ili jesmo li pravilno promijenili grupu, to ćemo postići naredbom `id`. Ona kao rezultat vraća UID i GID brojeve vezane za trenutnog korisnika te popis grupa kojima korisnik pripada. Argument ove naredbe može biti i neko drugo korisničko ime ukoliko želimo saznati na primjer kojoj grupi pripada neki drugi korisnik.

```
$ id
uid=501(luka) gid=501(luka) groups=501(luka)
```

2.9. Zadaci

1) Promijenite svoju zaporku (pažnja – nemojte ju zaboraviti!)

2) Odjavite se sa sustava i prijavite se opet pomoću novo stvorene zaporke.

3) Ispišite na zaslon trenutno prijavljene korisnike sustava.

4) Odaberite jednog od trenutno prijavljenih korisnika i ispišite detaljnije informacije o njemu.

5) Pokušajte promijeniti svoje osobne podatke.

6) Ispišite na zaslon imena programa koje trenutno koriste prijavljeni korisnici.

7) Ispišite na zaslon identifikatore za proizvoljno odabrane korisnike.

3. Rad s datotekama i direktorijima

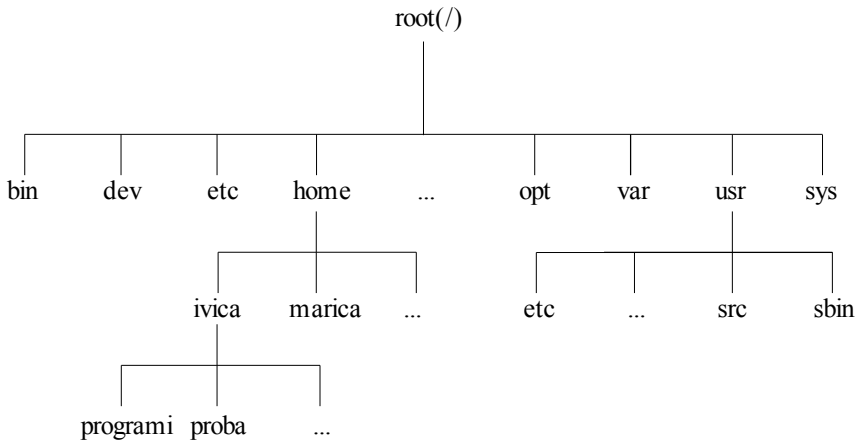
3.1. Uvod

Jezgra linuxa je tradicionalno raspolagala svim uređajima i svime ostalim kao datotekama. Kako su se tehnologije razvijale, nekim uređajima je postalo teško, pa i nemoguće raspolagati kao s datotekama u klasičnom smislu. Međutim, rad s datotekama je i dalje jedna od suštinskih vještina rada na Linuxu. Svaka od takvih datoteka ima svog vlasnika i neke interne informacije koje ju pobliže opisuju. Datoteke su hijerarhijski grupirane u direktorije (kazala). Direktorij je posebna datoteka koja sadrži druge datoteke i direktorije. U radu na sustavu potrebno je znati sadržaj direktorija, "šetati se" kroz direktorije, stvarati nove, brisati postojeće i drugo.

3.2. Hijerarhijska struktura datotečnog sustava

U Linuxu, kao i u svim operacijskim sustavima, također postoje datoteke i direktoriji (kazala). Ime datoteke sastoji se od jednostavnog niza slova, brojeva i interpunkcijskih oznaka. Linux omogućava ime datoteke u duljini od 256 znakova. S obzirom da je prijenos datoteka s jednog na drugo računalo vrlo bitan trebalo bi se držati ograničenja duljine imena datoteke od 14 znakova iz razloga što mogu nastati određeni problemi pri prijenosu datoteka na alternativne datotečne sustave.

Što se tiče FAT32 i NTFS datotečnih sustava ne bi trebalo biti problema po tom pitanju. Direktorij koji sadrži sve ostale direktorije u Linuxu je direktorij root koji se označava kosom crtom(*/),* kao što prikazuje slika 3.1



Slika 3.1.Struktura datotečnog sustava

Svaki korisnik operacijskog sustava Linux posjeduje svoj matični direktorij koji mu se dodijeli pri dodavanju korisnika na operacijski sustav. Prema dogovoru (odnosno prema Filesystem Hierarchy Standard - FHS, 1994) matični direktoriji svih korisnika nalaze se u direktoriju `/home`. Direktoriji u Linuxu su datoteke koje sadrže imena datoteka i poddirektorije te usmjerivače na te datoteke i poddirektorije. Ako se ispiše sadržaj nekog direktorija na ekran, ustvari se ispiše sadržaj datoteke-kazala. Kada se mijenja ime određene datoteke, a pri tome se datoteka nalazi u tekućem direktoriju, ustvari se mijenja zapis u datoteci-kazalu. Ako se određena datoteka premješta s jednog direktorija u drugi direktorij, to ustvari znači da se premješta njezin opis iz jedne datoteke-kazala u drugu. Svaka datoteka ima svoj broj (`inode`) koji fizički reprezentira njenu lokaciju na disku. Jednoj datoteci predstavljenoj jednim inode brojem, može se pristupati s različitim imenima. Tako se dobiva dojam da su to dvije različite datoteke, iako je njihov sadržaj isti.

Datoteke se u Linuxu dijele na četiri tipa: obične datoteke, datoteke-kazala, veze, posebne datoteke. Obične datoteke mogu sadržavati dokumente, osnovni kod u C jeziku, binarne izvršne datoteke, razne skripte itd. Datoteke-kazala objašnjene su prethodno. Veze u pravom smislu riječi nisu datoteke već elementi u kazalu koji su usmjereni na datoteke.

U svijetu u kojem živi i u okolini koja ga okružuje čovjek svakodnevno uočava oko sebe stvari i pojave koje su smještene na točno određenim mjestima i po propisanim pravilima. Jedan od općenitih primjera za takvo nešto je npr. fakultet. Fakultet je, radi lakšeg rada i snalaženja organiziran u više odjela (zavoda) tj. može se reći da fakultet posjeduje nekakvu hijerarhiju. Npr. ako student "Pero Perić" želi doći do dekana on zna da će se direktno uputiti na dekanat, a ne npr. na zavod za matematiku.

Analogno, na operacijskom sustavu Linux postoje različiti direktoriji ("odjeli") u kojima su smještene točno određene datoteke. Sustav je koncipiran na takav način iz prethodno rečenog razloga, a to je: jednostavniji rad, lakše snalaženje po sustavu itd.

3.3. Prikaz sadržaja direktorija

Na Linuxu se sadržaj direktorija doznaje korištenjem naredbe `ls`, koja podržava čitav niz parametara kojim prilagođavamo ispis našim potrebama. U DOS-u je to naredba `dir`, a kako bi na Linuxu dobili ekvivalentnu naredbu, trebamo koristiti naredbu s parametrima `ls -C -b`. Ostali parametri su objašnjeni u daljnjem tekstu.

Osnovnim korištenjem naredbe `ls` bez parametara dobivamo popis svih datoteka i direktorija, poredan abecednim redom, odozgor prema dolje te s lijeva na desno. Budući Linux razlikuje mala i velika slova, prednost velikog nad malim slovom uzima se u obzir prilikom sortiranja.

Ako želimo saznati više informacija, naredbi `ls` dodajemo parametre. Za prikaz veličine datoteke koristimo parametar `-s`:

Primjer:

```
$ ls -s
total 14          2 Mail      2 dat.txt         2 mail
   2 DeadLetters  2 atlas     2 dead.letter    2 write
```

Prikazani broj uz datoteku ili direktorij je količina blokova zaokružena na prvo cijelo. Linux radi sa blokovima podataka, a veličina bloka ovisi o datotečnom sustavu. Obično je 1 kilobajt, a ponekad 512 bajta. Fizička veličina se dobiva množenjem broja blokova sa veličinom bloka. Zbroj blokova je naveden pod **`total`,** ali on ne uključuje veličinu podređenih direktorija. Direktorij iz našeg primjera zauzima 14 * 1 kB, tj. 14 kilobajta.

Za informacije o veličinama pojedinih datoteka ili direktorija, iza naredbe `ls -s` navodimo njihova imena odvojena razmakom:

Primjer:

```
$ ls -s prog1.a prog2.a
2 prog1.a     2 prog2.a
```

Datoteke koje počinju sa točkom (.) Linux prepoznaje kao skrivene datoteke (ili direktorije). U takve

se datoteke u početnom direktoriju snimaju korisničke konfiguracije. Za prikaz takvih datoteka koristi se parametar `-a`:

Primjer:

```
$ ls -a
.                       .login                  .profile
..                      .logout                 .viminfo
.TTauthority            .mozilla                 DeadLetters
.Xauthority             .pine-debug1             Mail
.addressbook            .pine-debug2             atlas
.addressbook.lu         .pine-debug3             dat.txt
```

Početni direktorij pun je skrivenih datoteka.

Ako želimo detaljan izvještaj o datotekama koji uključuje i prikaz dozvola koristimo parametar `-l`. Možemo koristiti kombinaciju s parametrom `-a` da dobijemo izvještaj o svim pa tako i skrivenim datotekama.

Primjer:

```
$ ls -la
total 178
drwx------   8 marko    other       1024 Apr 29 00:47 .
drwxr-xr-x 110 root     other       2048 Aug 31  2004 ..
drwx------   2 marko    pripr        512 Apr 21 00:58 mail
----------   1 marko    pripr        179 Apr 12 23:44 write
```

O dozvolama će biti riječi kasnije, a sada spomenimo da postoje tri vrste dozvola za tri grupe korisnika. Svaka od njih sadrži dozvolu za čitanje "`r`" (engl. read), dozvolu za pisanje "`w`" (engl. write) i dozvolu za izvršavanje "`x`" (engl. execute). Slovo "`d`" na početku označava direktorij, a znak "`-`" datoteku. U prvom stupcu je navedeno radi li se o datoteci ili direktoriju, te vrste dozvola. U drugom stupcu je broj tvrdih poveznica datoteka (engl. hard link), o kojima će također nešto više biti rečeno kasnije. Potom slijede stupci s imenom vlasnika datoteke, `other` i `pripr` su korisničke grupe na pinusu. Završni stupac sadrži veličinu direktorija ili datoteke, te podatak o zadnjoj promjeni.

Lista parametara za naredbu `ls` je dugačka, a u tablici 3.1su prikazani već objašnjeni parametri uz dodatak još nekih koji se često koriste.

Tablica 3.1: Popis nekih oznka naredbe ls

Parametar	Značenje
-a	Ispisuje sve datoteke, uključivši i skrivene
-F	Navodi tip datoteke stavljajući oznake “/” za direktorij , “*” za izvršnu datoteku, “@” za simbolične veze
-s	Prikaz veličine datoteke u blokovima
-C	Prikaz u stupcima
-1	Prikaz u jednom stupcu
-l	Detaljan izvještaj
-g	Slično kao -l, samo bez prikaza vlasnika
-r	Prikaz u obrnutom redosljedu (engl. reverse)
-t	Sortiranje po vremenu modifikacije
-u	Sortiranje po vremenu pristupa
-S	Sortiranje po opadajućoj veličini datoteka
-d	Prikaz samog direktorija, ne njegovog sadržaja

3.4. Naredba pwd

Tijekom kretanja po direktorijima često puta korisnika zanima u kojem se direktoriju trenutno nalazi odnosno koja je staza direktorija u kojem se korisnik trenutno nalazi. Naredba koja omogućava pregled staze direktorija u kojem se korisnik nalazi u svakom trenutku je naredba `pwd`. Dakle, ako se korisnik nalazi trenutno u direktoriju `/ivica`, pokretanjem naredbe `pwd` na terminalu će se ispisati `/home/ivica`. Primjer:

```
$ pwd
/home
$ cd ivica
$ pwd
/home/ivica
```

3.5. Kreiranje novog direktorija

Kreiranje novog direktorija postižemo naredbom `mkdir` (engl. make directory) koja je vrlo jednostavna za korištenje. Treba se samo pozicionirati u direktorij u kojem želimo kreirati novi direktorij i navesti ime novog direktorija ili iz bilo kojeg drugog direktorija navesti apsolutni put. Apsolutni put se navodi počevši od korjenskog direktorija (engl. root), koje se označava kao "`/`", tako da se svaki sljedeći direktorij odvoji istim znakom "`/`", sve dok ne navedemo i posljednji, željeni direktorij.

Primjer:

```
$ ls
DeadLetters      SQL          atlas        mail
Mail             dead.letter  write
$ mkdir VJEZBA
$ ls
DeadLetters      SQL          VJEZBA       dead.letter  write
Mail             atlas        mail
```

Pogledajmo unutar novostvorenog direktorija:

Primjer:

```
$ ls -ld VJEZBA
drwx------   2 marko    pripr        512 Nov  1 22:32 VJEZBA
$ ls -la VJEZBA
total 4
drwx------   2 marko    pripr        512 Nov  1 22:32 .
drwx------  11 marko    other       1024 Nov  1 22:32 ..
```

Prvom naredbom dobili smo prikaz samog direktorija te njegovog detaljnog izvještaja. Pretpostavljene (*default*) dozvole postavljene su `umask` naredbom. Drugom naredbom smo zavirili unutar novostvorenog direktorija i pronašli samo specijalne datoteke "`.`" i "`..`". Jedna pokazuje na sebe samu, a druga na nadređeni direktorij.

Prilikom stvaranja novog direktorija treba imati na umu da su potrebne odgovarajuće dozvole u radnom direktoriju.

3.6. Promjena direktorija

Prijavom na sustav, korisnik se pozicionira u svoj matični direktorij (engl. home) kojeg je je najčešće potrebno promijeniti. To činimo naredbom `cd`**.**

Naredba `cd` (engl. change directory) je naredba kojom mijenjamo trenutni, odnosno radni direktorij. Novi direktorij zadajemo relativno radnom direktoriju ili apsolutnim putem. Oba načina prikazat ćemo primjerima.

Primjer:

```
$ pwd
/home/marko
$ cd VJEZBA
$ pwd
/home/marko/VJEZBA
```

Ovim primjerom mijenjamo radni direktorij u direktorij VJEZBA koje se nalazi u radnom direktoriju (ta činjenica nam omogućava korištenje relativnog puta). Naredbom `pwd` provjeravamo koji je trenutačni direktorij. Da bi se pozicionirali u direktorij VJEZBA iz bilo kojeg drugog direktorija, trebamo uz naredbu `cd` navesti apsolutni put.

Primjer:

```
$ pwd
/home/marko/VJEZBA
$ cd /home/marko/mail
$ pwd
/home/marko/mail
```

Navodeći apsolutni put, pozicioniramo se u navedeni direktorij bez obzira u kojem se direktoriju sada nalazili. Unosom naredbe `cd mail` sustav bi javio grešku. Greška bi se javila jer se u radnom direktoriju ne nalazi direktorij mail. U zadavanju puta se možemo koristiti specijalnim direktorijem "`..`" koji pokazuje na nadređeni direktorij.

Primjer:

```
$ pwd
/home/marko/mail
$ cd ../VJEZBA
$ pwd
/home/marko/VJEZBA
```

U putu možemo zadavati serije "`..`" direktorija odvojenih znakom "`/`", što nam omogućava da s manje tipkanja dođemo do istog rezultata.

`cd` bez parametara pozicionira nas u početni direktorij (engl. Home, koji je određen varijablom okoline HOME), a `cd /` nas pozicionira u izvorni sustavni direktorij ili korijen (engl. root). Korijen možemo zamisliti kao deblo drveta iz kojega se granaju sve grane (direktoriji) i listovi (datoteke).

Česta greška prijelaznika iz DOS-a je da upisuju `cd..` (bez razmaka). Treba upisati `cd ..` (s razmakom) jer je `cd` program, a "`..`" ispravni argument.

3.7. Kopiranje datoteka

Ukoliko želimo imati više kopija iste datoteke, koristimo naredbu `cp`.

Sintaksa naredbe sadrži izvorišnu datoteku koju slijedi odredišna datoteka. Ciljna i odredišna datoteka mogu biti određene apsolutnom ili relativnom stazom do datoteke ili direktorija.

Naredbom `touch` (vidi 5. poglavlje) stvoriti ćemo za potrebe ovog primjera datoteku "tekst.txt". Naredbu `cp` slijedi datoteka koju želimo kopirati te datoteka u koju želimo kopirati sadržaj.

Primjer:

```
$ touch tekst.txt
$ cp tekst.txt tekst.copy
$ ls -l tekst.txt tekst.copy
-rw-------   1 marko    pripr          0 Nov  1 23:06 tekst.copy
-rw-------   1 marko    pripr          0 Nov  1 23:06 tekst.txt
```

Iz detaljnog izvještaja primjećujemo da su se dozvole također kopirale.
Ako u sintaksi naredbe uz datoteku (ili seriju datoteka) slijedi ime direktorija, tada će se navedena(e) datoteka(e) kopirati u navedeni direktorij i to s imenom (ili serijom imena) preuzetim od originalne datoteke.

Primjer:

```
$ mkdir NOVO
$ cp tekst.copy tekst.txt NOVO
$ ls -l NOVO
total 4
-rw-------   1 marko    pripr          0 Nov  1 23:17 tekst.copy
-rw-------   1 marko    pripr          0 Nov  1 23:17 tekst.txt
```

Postupak kopiranja sadržaja direktorija sastoji se od stvaranja novog direktorija i kopiranja sadržaja iz izvorišnog direktorija u novostvoreni direktorij.

Želimo li kopirati direktorij sa svim njegovim sadržajem u novi direktorij, naredbi `cp` dodajemo parametar `-r`.

Primjer:

```
$ ls NOVO
tekst.copy  tekst.txt
$ cp -r NOVO NOVO2
$ ls NOVO2
tekst.copy  tekst.txt
```

Vidljivo je da su se datoteke kopirale u novostvoreni direktorij.

3.8. Premještanje i preimenovanje datoteka

Radnje vezane uz preimenovanje i premještanje datoteka na Linux sustavu radimo jednom naredbom. To je naredba `mv` (engl. move), koja služi za premještanje datoteke u neki drugi direktorij ili promjenu imena datoteke (direktorija). Oblik pisanja naredbe je gotovo identičan naredbi `cp`, a time i njezina pravila korištenja. Po želji odabran broj datoteka možemo premjestiti iz jednog direktorija u drugo, uz napomenu da kao korisnik moramo imati dozvolu pisanja za taj direktorij.

Korištenjem naredbe `cp` za kopiranje direktorija (ili datoteke) potreban je parametar `-r`, što nije slučaj i za naredbu `mv`.

Želimo direktoriju NOVO promijeniti ime u STARO. To ćemo pokazati na primjeru.

Primjer:

```
$ ls NOVO
tekst.copy  tekst.txt
$ mv NOVO STARO
$ ls STARO
tekst.copy  tekst.txt
```

Slično ćemo postupiti ako želimo premjestiti direktorij na novu poziciju, npr. u nadređeni direktorij.

Primjer:

```
$ mv STARO ../
$ ls ../
DeadLetters      SQL         atlas       mail
Mail             STARO       VJEZBA      dead.letter  write
```

3.9. Brisanje datoteke

Naredba `rm` (engl. remove) je naredba koju koristimo za brisanje datoteka. Potrebno je biti jako oprezan s naredbom `rm` zato što se, za razliku od DOS-a, jednom obrisana datoteka ne može više povratiti. Ovom naredbom ne možemo brisati direktorije, osim ako joj ne dodamo parametar `-r` (engl. recursive) kada brišemo direktorij i sve pripadne datoteke (tj. datoteke koje se nalaze u tom direktoriju). Parametar `-i` (engl. interactive) služi kao mjera sigurnosti na način da traži potvrdu (`yes, no`) za brisanje svake datoteke posebno.

Primjer:

```
$ ls -l
total 6
drwx------   2 marko    pripr        512 Nov  1 23:22 NOVO2
-rw-------   1 marko    pripr          0 Nov  1 23:06 tekst.copy
-rw-------   1 marko    pripr          0 Nov  1 23:06 tekst.txt
$ rm -i tekst.copy tekst.txt
rm: remove tekst.copy (yes/no)? Y
rm: remove tekst.txt (yes/no)? N
$ ls -l
total 4
drwx------   2 marko    pripr        512 Nov  1 23:22 NOVO2
-rw-------   1 marko    pripr          0 Nov  1 23:06 tekst.txt
```

Naredba `rm -i` će za sve datoteke navedene u sintaksi naredbe tražiti potvrdu brisanja. Vidimo da je nakon brisanja u trenutnom direktoriju ostala datoteka "tekst.txt" jer ona nije dobila potvrdu.

Primjer:

```
$ cd ..
$ ls -lR VJEZBA
VJEZBA:
total 4
drwx------   2 marko    pripr        512 Nov  1 23:22 NOVO2
-rw-------   1 marko    pripr          0 Nov  1 23:06 tekst.txt
VJEZBA/NOVO2:
total 4
-rw-------   1 marko    pripr          0 Nov  1 23:22 tekst.copy
-rw-------   1 marko    pripr          0 Nov  1 23:22 tekst.txt
$ rm -r VJEZBA
$ ls -lR VJEZBA
VJEZBA: No such file or directory
```

U ovom "opasnom" primjeru pokazali smo pravu snagu naredbe `rm` korištenjem parametra `-r`. Naredba `ls -lR` nam, također rekurzivno, izlistava direktorij i svaki njegov podređeni direktorij (VJEZBA/NOVO2). Nakon rekurzivnog brisanja je sve što se nalazilo unutar direktorija VJEZBA

(sve datoteke i direktoriji) nepovratno uništeno.

3.10. Brisanje direktorija

Za brisanje direktorija koristimo naredbu `rmdir` (engl. remove directory). Uvjet da bi mogli obrisati neki direktorij svakako je dozvola za pisanje i čitanje. Drugi uvjet koji je potreban prije brisanja direktorija je da on mora biti prazan, tj. u njemu ne smije biti nikakvih datoteka. Time se donekle sprečava neželjeni gubitak sadržaja.

Pokazat ćemo to na primjeru.

Primjer:

```
$ ls -l STARO
total 4
-rw-------   1 marko    pripr          0 Nov  1 23:29 tekst.copy
-rw-------   1 marko    pripr          0 Nov  1 23:29 tekst.txt
$ rmdir STARO
rmdir: directory "STARO": Directory not empty
```

Pokušali smo obrisati direktorij koji nije prazan te nam je sustav javio grešku. Ako prethodno obrišemo sadržaj, možemo uspješno obrisati i direktorij.

Primjer:

```
$ rm STARO/*
$ rmdir STARO
$ ls -l STARO
STARO: No such file or directory
```

Korištenjem zamjenskih znakova (8. poglavlje) obrisali smo sadržaj direktorija STARO, a nakon toga i sâm direktorij.

3.11. Zadaci

1) U korjenskom direktoriju stvorite novi direktorij “Vjezba”

2) Pomoću naredbe `touch` upišite neki tekst u datoteku “tekst.txt”

3) U direktoriju “Vjezba” stvorite novi direktorij “Backup” i kopirajte tamo datoteku “tekst.txt”

4) Novokopiranu datoteku preimenujte u “tekst.copy”

5) Izlistajte rekurzivno sve direktorije sa punim izvještajem iz direktorija “Vjezba”

6) Pokušajte obrisati direktorij “Backup” koristeći naredbu `rmdir`. Što se dogodilo?

7) Obrišite koristeći naredbu `rm` direktorij “Backup”

8) Uvjerite se da direktorija više nema

9) U direktoriju “Vjezba” obrišite datoteku “tekst.txt”

10) Iz korjenskog direktorija koristeći naredbu `rmdir` obrišite direktorij “Vjezba”

4. Struktura datotečnog sustava u Linuxu

4.1. Uvod

Klasičan sastav direktorija u operativnom sustavu Linux sastoji se od direktorija: `/bin`, `/dev`, `/etc`, `/lib`, `/lost+found`, `/mnt`, `/sys`, `/tmp`, `/usr`. Napisani direktoriji nalaze se u `root(/)` direktoriju i oni su neophodni za rad operacijskog sustava Linux. Naravno da to nisu svi direktoriji na operativnom sustavu Linux, oni ovise o broju i vrsti paketa koji su instalirani na sustav.

4.2. Direktorij `/bin`

U direktoriju `/bin` se nalaze svi programi koji su potrebni za pokretanje sustava kao i većina naredbi koje Linux koristi. Pod pojmom naredbe misli se na binarne datoteke u ovom direktoriju koje su u stvari izvršne datoteke. Tako se, izvođenjem naredbe `ls` u direktoriju `/bin`, između ostalih izlistaju i sljedeće naredbe: `cat`, `gunzip`, `ping`, `cp`, `mv` itd. Radi se o izvedbenim naredbama, koje se nalaze i u sljedećim direktorijima: */usr/ucb,* te */usr/bin*.

4.3. Direktorij `/dev`

U kazaSlu `/dev` nalaze se specijalne datoteke koje mogu, ali i ne moraju odgovarati nekom upravljačkom programu u jezgri sustava. Kako se u Linuxu sve temelji na datotekama, svaka jedinica (disketna jedinica, tvrdi disk, USB (serijiski US random pristup) , COM portovi itd.) priključena na računalo, određena je specijalnim datotekama koje se nalaze u ovom direktoriju. Glavna podjela između jedinica je na takozvane "*blokovne uređaje"* i "*znakovne uređaje"*. Generalno "blokovni uređaji*"* predstavljaju jedinice koje pohranjuju ili čuvaju podatke (npr. tvrdi disk, disketna jedinica itd.) dok se "znakovni uređaji*"* mogu shvatiti kao jedinice koje prenose podatke (npr. Serijski port, paralelni port, miš itd.). Ilustracije radi, slijedi primjer dijela sadržaja `/dev` direktorija:

```
$ ls -al
crw-------     1 jelena root  14,   4 Oct 24  2005 audio
crw-------     1 jelena root  14,  20 Oct 24  2005 audio1
lrwxrwxrwx     1 root   root        3 Oct 24  2005 cdrom -> hdc
lrwxrwxrwx     1 root   root        3 Oct 24  2005 cdwriter -> hdc
crw-------     1 jelena root   5,   1 Oct 24  2005 console
lrwxrwxrwx     1 root   root        3 Oct 24  2005 dvd -> hdc
lrwxrwxrwx     1 root   root        3 Oct 24  2005 floppy -> fd0
```

Datoteke `/dev/cdrom` i `/dev/fd0` predstavljaju redom CDROM i disketnu jedinicu. Važno je napomenuti da su navedene datoteke iste kao i sve ostale što znači da se njih može upisivati i iz njih čitati. Ako se uzme za primjer datoteka `/dev/dsp` koja predstavlja interni zvučnik u računalu upisivanjem predviđenog teksta u tu datoteku interni zvučnik u računalu će reproducirati nekakav zvuk.

Ako se izlista sadržaj direktorija `/dev` vide se datoteke `hda`, `hda1`, `hda2` itd. Te datoteke predstavljaju glavni disk (`hda`) i različite particije glavnog diska (`hda1`,`hda2`..):

```
$ ls -al
brw-r-----     1 root   disk   3,   0 Oct 24  2005 hda
brw-r-----     1 root   disk   3,   1 Oct 24  2005 hda1
brw-r-----     1 root   disk   3,   2 Oct 24  2005 hda2
brw-r-----     1 root   disk   3,   3 Oct 24  2005 hda3
```

Od ostalih datoteka u daljnjem tekstu spomenute su one koje se najčešće spominju u praksi, a prikazane s u tablici 4.1

Tablica 4.1: Ostale datoteke direktorija /dev

Datoteka	Značenje
/dev/ttyS0 (COM1)	Prvi serijski port
/dev/psaux (PS/2)PS/2	Veza za miš ili tipkovnicu
/dev/lp0 (LPT1)	Prvi paralelni port
/dev/dsp	Prva audio jedinica
/dev/usb (USB)	Veza za USB jedinice
/dev/sda (C:\, SCSI)	Memorijske jedinice(stickovi) itd.
/dev/scd (D:\, CD-ROM)	Prvi CDROM

4.4. Direktorij `/etc`

U direktoriju `/etc` i u njegovim poddirektorijima nalaze se sve konfiguracijske datoteke. Konfiguracijska datoteka definira se kao datoteka s kojom se kontrolira rad određenog programa.

Neke osnovne konfiguracijske datoteke su: *`/etc/passwd`, `/etc/group`, `/etc/networks`, `/etc/lilo.conf`,`/etc/modules`,`/etc/profile`*

Interakcija između programa i konfiguracijske datoteke se odvija na način da program čita postavke iz datoteke i na temelju toga usmjerava svoje izvršavanje. Linije u datoteci koje počinju znakom # se zanemaruju od strane programa te program koristi samo one linije u datoteci koje ne počinju s #. Kao što je rečeno u direktoriju `/etc` nalaze se sve konfiguracijske datoteke, a od bitnijih valja spomenuti datoteke `passwd` i `group` u kojima su redom spremljene lozinke korisnika sustava te podaci o tome kojoj grupi koji korisnik pripada. Slijedi primjer dijela sadržaja direktorija `/etc`:

```
$ ls -al
-rw-r--r--   1 root root     236 May 22 21:14 eclipse.conf
drwxr-xr-x   2 root root    4096 Sep  1 13:00 fonts
-rw-r--r--   1 root root     594 Sep  1 13:34 group
-rw-r--r--   2 root root     152 Oct 20 13:33 hosts
drwxr-xr-x   3 root root    4096 Sep  1 13:05 mail
drwxr-xr-x   2 root root    4096 Jul 24 06:02 mplayer
-rw-r--r--   1 root root    1539 Sep  1 13:34 passwd
-rw-r--r--   1 root root     840 Apr 15  2005 profile
-rw-r--r--   2 root root      24 Oct 20 16:34 resolv.conf
drwxr-xr-x   4 root root    4096 Oct 21 11:58 vmware
drwxr-xr-x   2 root root    4096 Sep  1 13:01 yum
-rw-r--r--   1 root root     253 May 25 19:46 yum.conf
```

4.5. Direktorij `/lib`

U direktoriju `/lib` nalaze se sve dijeljene (shared) i statičke (static) *biblioteke* koje programi koriste. Biblioteke su u stvari datoteke te ih se može promatrati kao nekakve potprograme koje određeni

program poziva kad mu zatrebaju, npr. potprograme za programski jezik C i druge programske jezike. Neke od njih se pozivaju kad ih određeni program treba dok su neke konstantno aktivne jer ih sustav konstantno koristi. Nastavak imena datoteka koje predstavljaju biblioteke na operativnom sustavu Linux je `*.so` što je ekvivalent `*.dll` na Windows operativnom sustavu. Slijedi primjer dijela sadržaja direktorija `/lib`:

```
$ ls -al
-rwxr-xr-x          1 root root   22624  Oct 28  2004    libnss_dns-
2.3.3.so
lrwxrwxrwx          1 root root   14     Sep  1  13:47   cpp ->
../usr/bin/cpp
lrwxrwxrwx          1 root root   19     Apr 28 18:08    libnss_dns.so.2
-> libnss_dns-2.3.3.so
-rwxr-xr-x          1 root root   47244  Oct 28  2004    libnss_files-
2.3.3.so
lrwxrwxrwx          1 root root   18     Apr 28 18:08
libnss_files.so.1 -> libnss1_files.so.1
lrwxrwxrwx          1 root root   21     Apr 28 18:08
libnss_files.so.2 -> libnss_files-2.3.3.so
-rwxr-xr-x          1 root root   23528  Oct 28  2004    libnss_hesiod-
2.3.3.so
-rwxr-xr-x          1 root root   281840 Aug 31 2004    libnss_ldap-
2.3.3.so
lrwxrwxrwx          1 root root   20     Apr 28 18:10
libnss_ldap.so.2 -> libnss_ldap-2.3.3.so
```

4.6. Direktorij `/lost+found`

Direktorij `/lost+found` vezan je uz postupke oporavka cijelog datotečnog sustava. U njega se pohranjuju spašeni fragmenti datoteka koje imaju izgubljena imena tj. imaju samo inode broj. Inode broj je broj pridjeljen svakoj datoteci i jednoznačno ju označava. Inode brojevi datoteka mogu se vidjeti izvođenjem naredbe `ls -i`. U pravilu direktorij `/lost+found` ne bi trebalo sadržavati nikakve datoteke osim ako prethodno nije došlo do naglog pada sustava (npr. zbog nestanka struje).

4.7. Direktorij `/mnt`

Direktorij `/mnt` se najčešće koristi za montiranje lokalnih ili udaljenih datotečnih sustava. Naime prije korištenja određenog datotečnog sustava na Linuxu (npr. CDROM-a) datotečni sustav treba biti prethodno montiran te se taj postupak, koji će u narednim poglavljima biti objašnjen, odvija u ovom direktoriju.

4.8. Direktorij `/var`

Direktorij `/var` sadrži datoteke bitne za rad sustava, dijela sustava ili nekog programa. Na primjer, u direktoriju `/var/spool/mail` se najčešće nalazi e-pošta korisnika na sustavu smještena u odgovarajućem poddirektoriju s nazivom korisnika. Uglavnom se podacima iz ovog direktorija pristupa automatski, tokom rada samih programa. Najčešće programi ovdje imaju poddirektorije u koje smještaju svoje podatke. Na primjer za email, gdje programi za razmjenu pošte koriste

spomenuti poddirektorij. Tako se u `/var/spool` nalaze poddirektoriji s podacima koji čekaju u redu da budu negdje premješteni, poslani, ili isprintani (ovisno o postavkama sustava). Ispis sadržaja direktorija `/var`:

```
$ ls -al
drwxr-xr-x   2 root    root    4096 Sep  1 13:00 account
drwxr-xr-x   7 root    root    4096 Sep  1 13:04 cache
drwxr-xr-x   3 netdump netdump 4096 Aug 22 21:13 crash
drwxr-xr-x   2 root    root    4096 Aug 23 10:44 cvs
drwxr-xr-x   3 root    root    4096 Sep  1 13:00 db
drwxr-xr-x   3 root    root    4096 Sep  1 13:01 empty
drwxrwx--T   2 root    gdm     4096 Oct 24 20:48 gdm
drwxr-xr-x  19 root    root    4096 Sep  1 13:31 lib
drwxr-xr-x   2 root    root    4096 May 23 06:28 local
drwxrwxr-x   4 root    lock    4096 Oct 24 20:47 lock
drwxr-xr-x   9 root    root    4096 Oct 24 20:47 log
lrwxrwxrwx   1 root    root      10 Sep  1 12:57 mail -> spool/mail
drwxr-x---   4 root    named   4096 Aug 22 22:33 named
drwxr-xr-x   2 root    root    4096 May 23 06:28 nis
drwxr-xr-x   2 root    root    4096 May 23 06:28 opt
drwxr-xr-x   2 root    root    4096 May 23 06:28 preserve
drwxr-xr-x   2 root    root    4096 Mar 28  2005 racoon
drwxr-xr-x  16 root    root    4096 Oct 24 20:47 run
drwxr-xr-x  13 root    root    4096 Sep  1 13:01 spool
drwxrwxrwt   3 root    root    4096 Oct 21 11:48 tmp
```

4.9. Direktorij `/sys`

Direktorij `/sys` koristi jezgra (kernel) kao virtualni datotečni sustav tj. kao sučelje prema informacijama iz jezgre sustava. Ispis sadržaja direktorija `/sys`:

```
$ ls -al
drwxr-xr-x  12 root root    0 Oct 24 20:47 bus
drwxr-xr-x  24 root root    0 Oct 24 20:47 class
drwxr-xr-x   7 root root    0 Oct 24  2005 devices
drwxr-xr-x   3 root root    0 Oct 24  2005 firmware
drwxr-xr-x   3 root root    0 Oct 24  2005 kernel
drwxr-xr-x  67 root root    0 Oct 24 20:47 module
drwxr-xr-x   2 root root    0 Oct 24  2005 power
```

4.10. Direktorij `/tmp`

Direktorij `/tmp` služi za privremenu pohranu podataka. Obično se sve što se nalazi u ovom direktoriju pobriše prilikom restartiranja sustava, ali ne nužno (ovisno o postavkama sustava).

4.11. Direktorij `/usr`

Direktorij `/usr` obično sadrži najviše podataka na sustavu. Ono u stvari predstavlja novi virtualni sustav na postojećem sustavu. Tu se smješteni razni korisnički programi (`telnet`, `ftp`..*)* , igre,

dokumenti, programi za administriranje sustava, uključujuće datoteke koje su potrebne pri stvaranju izvršnih (binarnih) datoteka itd. Uz gubitak podataka iz direktorija `/usr` sustav bi i dalje morao moći obavljati one osnovne funkcije. Direktorij `/usr` u sebi sadrži nekoliko značajnijih poddirektorija kao što su `/bin`, `/sbin`, `/share`, `/man`. Slijedi opis navedenih poddirektorija:

4.12. Direktorij `/usr/bin`

U ovom direktoriju nalaze se izvršne (binarne) datoteke različitih korisničkih programa kao što su `vi` i `gcc`.

```
$ ls -al
-rwxr-xr-x   1 root root    2333416 Aug 10 16:35 vim
-rwxr-xr-x   2 root root     107036 Jul 27 13:50 gcc
```

4.13. Direktorij `/usr/sbin`

U ovom direktoriju nalaze se programi koji služe za administriranje sustava s tim da se njima služi isključivo administrator sustava.

4.14. Direktorij `/usr/share`

Direktorij `/usr/share` sadrži razne datoteke (dokumenti, ikone, slova itd..) koje mogu koristiti svi korisnici na sustavu. Taj direktorij može se shvatiti i kao direktorij za distribuciju datoteka među korisnicima sustava. Neke od datoteka su: fonts, `doc`, `icons`, `file`, `games`, `services.`

4.15. Direktorij `/usr/man`

Unutar direktorija `/usr/man` nalaze se informacije u kojima je opisan način rada određenih dijelova sustava, programa, zabavnih aplikacija itd. Tu se također mogu pronaći i tekstovi koji opisuju određene vrste datoteka na sustavu, način rada sustava u cijelini itd. Sam direktorij je organiziran tako da se unutar njega nalazi osam cjelina te svaka od tih cjelina u svom opisu obuhvaća jedan dio gore navedenog.

4.16. Specijalni direktoriji `.` i `..`

Direktorij `.` pokazuje na trenutnu poziciju određenog direktorija u hijerarhiji. Direktorij `..` pokazuje na poziciju prethodnog direktorija. (npr. Ako se nalazimo u direktoriju `/ivica` tada se i direktoriji `..` i `.` nalaze u direktoriju `/ivica` samo što "." pokazuje na direktorij `/ivica`, a direktorij `..` pokazuje na direktorij `/home`. Sada je jasno zašto se s naredbom `cd ..` (change directory) vraćamo u prethodni direktorij.

4.17. Početni direktorij

Početni direktorij različit je za svakog korisnika i obično nosi ime po "username-u" korisnika. Nakon što se određeni korisnik prijavi na sustav, sustav automatski korisniku otvara njegov početni direktorij. Početni direktoriji svih korisnika nalaze se prema Filesystem Hierarchy Standardu - FHS,

1994 u direktoriju `/home`. Logično je da korisnik koji se prijavi na sustav može mijenjati samo sadržaj svog matičnog (u ovom slučaju početnog) direktorija dok ostale direktorije može čitati ukoliko je to dozvoljeno.

4.18. Nadgledanje zauzeća diskovnog prostora

Svakom korisniku na operativnom sustavu Linux dodijeljuje se određeni dio diskovnog prostora. Vrlo je važno što ekonomičnije raspodijeliti diskovni prostor iz više razloga (npr. obično administrator sustava ne zna koliko će točno korisnika imati pristup sustavu; što je više korisnika, proporcionalno se smanjuje količina diskovnog prostora po svakom korisniku; treba osigurati dovoljan dio diskovnog prostora za nadogradnju korisničkih aplikacija na sustav...itd.). Kako za korisnike, tako je i za administratora korisno znati trenutno zauzeće diskovnog prostora na računalu. Administrator je taj koji treba voditi brigu o zauzeću cjelokupnog diskovnog prostora, dok obični korisnik kao takav treba znati koliko mu je dodijeljeno diskovnog prostora na korištenje i koliko je trenutačno tog prostora zauzeto. Naredbe koje omogućuju pregled slobodnog i iskorištenog diskovnog prostora su: `du`, `df`, `quota`.

4.18.1. Naredba `du`

Naredba `du` ispisuje veličinu u KB(kilobyte) i imena svih direktorija u trenutnom direktoriju te koliko diskovnog prostora zauzima pojedini direktorij. Ako se želi doznati zauzeće diska svih direktorija u trenutnom direktoriju, a da se ne ispišu njihova imena, tada se piše `du -s`. Ukoliko se želi dobiti podatak o veličini svih datoteka koje se nalaze u trenutnom direktoriju i njegovim poddirektorijima tada se piše `du -a`. Slijedi primjer naredbe `du`:

```
$ du
32      ./.vmware
8       ./.Trash/untitled folder
74980   ./.Trash
24      ./.mozilla/firefox/d2uvyh0d.default/chrome
12552   ./.mozilla/firefox/d2uvyh0d.default/Cache
32      ./.adobe/Acrobat/7.0/Cache
48      ./.mplayer
215644  ./devil-linux-1.2.6-i686-SMP
5216    ./.openoffice.org2.0
```

4.18.2. Naredba `df`

Naredba `df` govori koliko ima ukupno diskovnog prostora na računalu i koliko je od tog iskorišteno a koliko neiskorišteno te daje podatak o iskorištenosti brojčano i u postotcima. Slijedi primjer naredbe `df`:

```
$ df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/hda3             11680552   5121552   5956092  47% /
/dev/shm                253724         0    253724   0% /dev/shm
/dev/sda1               255708     45676    210032  18% /media/usbdisk
```

4.18.3. Naredba `quota`

Naredba `quota` ispisuje zauzeće diska i veličinu diskovnog prostora predviđenog za određenog korisnika ili grupu.

Sintaksa naredbe je:

```
quota [-g group] [-u username].
```

Ako se napiše npr. `quota -g ekipa` ispisat će se podaci o zauzeću diska i veličini diskovnog prostora za grupu Ekipa, a ako se Npr. Napiše `quota -u ivica` ispisat će se podaci o zauzeću diska i veličini diskovnog prostora za korisnika Ivicu.

4.19. Zadaci

1) Pristupite svom matičnom direktoriju te ispišite sve datoteke koje ono sadrži.

2) Otkrijte koliko diskovnog prostora je predviđeno za vašu grupu i vas kao korisnika.

3) Otkrijte koliko diskovnog prostora trenutno zauzimaju datoteke u vašem matičnom direktoriju.

4) Otkrijte koliko trenutno ima particija glavnog diska na računalu.

5) Otkrijte trenutnu stazu u kojoj se nalazite.

5. Kreiranje datoteka

5.1. Naredba `touch`

Svaka datoteka u Linuxu koristi tri oznake vremena i datuma: datum nastanka datoteke, datum zadnje promjene i datum zadnjeg pristupa datoteci. Ako pretražujemo datoteke naredbom `find` na način da pretražujemo samo one datoteke koje su nastale u određenom vremenskom razdoblju ili su u nekom vremenskom razdoblju mijenjane, onda je potrebno datume pravilno ažurirati. Želimo li ažurirati vrijeme posljednje promjene datoteke, a da pri tome ne mijenjamo sadržaj datoteke, možemo to učiniti pomoću naredbe `touch`. Osnovna naredba `touch`, bez opcija izvorno je podešena da ažurira vremenske oznake tako da im pridjeljuje trenutno vrijeme, a ako datoteka ne postoji, naredba `touch` ju stvara. Naredba `touch` nam je korisna npr. ako sustav pokrene naredbu za sigurnosnu pohranu datoteka koje su nastale u nekom vremenskom razdoblju, a korisnik želi pohraniti i neke od datoteka koje su nastale izvan tog vremenskog razdoblja, onda je moguće pomoću naredbe `touch` uvrstiti odabrane datoteke u sigurnosnu pohranu (backup).

Naredbi `touch` moguće je dodati neke opcije ako želimo izmjeniti njezino osnovno ponašanje. Osnovne opcije naredbe prikazane su u tablici 5.1

Tablica 5.1: Osnovne opcije naredbe touch

Znak	Značenje
-a	Ažurira vrijeme pristupa datoteci
-m	Ažurira vrijeme mijenjanja datoteke
-c	Onemogućava stvaranje datoteke ako ona ne postoji
-d <*vrijeme*>	Postavlja odabrano vrijeme, a ne trenutno.*Vrijeme* može biti u nekoliko formata, može sadržavati imena mjeseci, vremenskih zona..
-r <*datoteka1*>	Upisuje stanje vremenskih oznaka datoteke1 u stanje vremenskih oznaka odabrane datoteke

Osnovni oblik naredbe touch:

```
touch -am <popisdatoteka>.
```

5.2. Naredba `file`

Datoteke u Linuxu za razliku od datoteka u DOS-u ne koriste ekstenziju iz kojeg se može pročitati o kojem tipu datoteke se radi. Naredba `file` je naredba koja na lagan način nudi uvid u tip datoteke. Nažalost, nareba file ponekad daje neprecizne rezultate. Pri određivanju tipa datoteke naredba file pregledava ime datoteke, analizira prvih nekoliko redaka datoteke i analizira dozvolu datoteke i iz tih podataka zaključuje o kojem se tipu datoteke radi. Problem se javlja ako npr. tekstualna datoteka započinje kao C program, vrlo lako je moguće da će `file` zaključiti da se radi o izvršnoj datoteci, a ne o tekstualnoj datoteci.

Za identifikaciju tipova datoteka `file` koristi `magic file` koji je obično smješten na lokaciji `/usr/share/magic` ili `/usr/share/misc/file/magic`, ovisno o distribuciji.

Osnovni oblik naredbe file:

```
file [opcije] <datoteke>
```

Primjer:

Želimo odrediti kojeg su tipa datoteke u direktoriju `vjezba.` Da bi smo to postigli potrebno je pozicionirati se u taj direktorij.

```
$ cd vjezba
$ ls
proba
src/
```

Nakon što navedemo naredbu `file` potrebno je navesti ime datoteke kojoj želimo saznati tip.

```
$ file proba
proba: ascii text
$ file src
src:  directory
```

Naredba `file` prepoznaje datoteku `proba` kao tekstualni dokument, a `src` kao direktorij.

5.3. Naredba `cat`

Želimo li pregledati sadržaj datoteke, jedna od naredbi koja nam stoji na rspolaganju je naredba `cat.` Naredba `cat` kao ulaz uzima popis datoteka ili jednu datoteku i ispisuje sadržaj. Kratica cat dolazi od riječi concatenate-spojiti, pa naredba `cat` spaja više datoteka s popisa i ispisuje ih zajedno na zaslonu brzinom kojom se slova mogu ispisivati na zaslonu. Ukoliko se radi o većoj datoteci moguće je s tipkama ^S i ^Q privremeno zaustaviti brzi ispis. Ako naredbi `cat` dodamo opciju -v, naredba će moći ispisati sve kontrolne znakove unutar neke datoteke. Na taj način će se moći ispisati čak i izvršne datoteke. Naredba `cat` je nezgodna za pregled sadržaja velikih datoteka pa se u tu svrhu koriste narebe `less` ili `more`.

Primjer:

Želimo ispisati sadržaj daototeka tekst i proba koje se nalaze u direktoriju vježba jedan iza drugoga. To postižemo slijedećom naredbom:

```
$ cat tekst proba
```

Naredba ispisuje sadržaj datoteka tekst i proba jedan iza drugog.

Ukoliko želimo ispisati sadržaj sistemske datoteke `/etc/passwd` možemo to učiniti slijedećom naredbom:

```
$ cat  /etc/passwd
```

U ovom primjeru se najbolje vidi nedostatak naredbe `cat`. Naredba će ispisati datoteku, ali će to

učiniti velikom brzinom pa je nezgodna za ispisivanje velikih datoteka.

5.4. Naredba `head`

Naredba `head` se koristi da bismo mogli vidjeti prvih n redaka datoteke.

Osnovni oblik naredbe `head`:

```
head <imedatoteke>.
```

Naredba u ovom obliku prikazuje prvih 10 redaka datoteke.

Oblik datoteke za proizvoljan broj redaka:

```
head -n  <brojredaka>  <imedatoteke>
```

pri čemu broj redaka može biti i pozitivan i negativan.

Neka se u direktoriju vjezba nalazi i datoteka tekst. Želimo pogledati prvih nekoliko redaka datoteke. To postižemo slijedećom naredbom:

```
$ head -n 15 tekst
```

Naredba ispisuje prvih petnaest redaka datoteke tekst.

Ako se u direktoriju nalaze dvije datoteke i želimo pogledati prvih nekoliko redaka obje datoteke koristimo slijedeću naredbu.

```
$ head -n 3 proba tekst
```

Naredba ispisuje prva tri retka datoteke proba, a ispod nje ispisuje prva tri retka datoteke tekst.

5.5. Naredba `tail`

Za razliku od naredbe `head`, naredba `tail` omogućuje pogled u zadnjih n redaka datoteke. Kao i `head`, `tail` koristi istu sintaksu i na isti način se upisuje proizvoljan broj linija koje želimo ispisati. Kombiniranjem naredbi `tail` i `head` moguće je vidjeti određene dijelove datoteka u Linuxu.

Primjer:

Želimo ispisati zadnjih nekoliko redaka datoteke tekst.

```
$ tail -n 15 tekst
```

Ispisuje zadnjih 15 redaka datoteke tekst.

5.6. Naredba `more`

Naredba `more` je vrlo korisna za pregledavanje velikih datoteka jer ispisuje datoteku stranicu po

stranicu. Naredba prepoznaje veličinu ekrana na kojem ispisuje sadržaj i prilagođava ispis. Broj redaka koji se može ispisati na ekranu ovisi o stanju varijable TERM. Naredba `more` je nastala u Berkeleyevoj inačici UNIX-a te se pokazala toliko korisnom da je postala standard u Linuxu kao i vi editor.

Osnovni oblik naredbe `more`:

```
more <imedatoteke>.
```

Naredba u osnovnom obliku ispisuje sadržaj datoteke na zaslonu, a na slijedeću stranicu datoteke se prelazi pritiskom tipke <Space>, a na slijedeći redak pritiskom tipke <Enter>. Za pregledavanje sadržaja većeg broja datoteka (`more <datoteka1 datoteka2 datoteka3>`) gdje je potrebno jednu od datoteka uređivati treba koristiti naredbu e ili v. Pritiskom na tipku <e> pokreće se program za uređivanje datoteka koji je za pojedinu datoteku određen varijablom $EDITOR, pritiskom na tipku <v> pokreće se program za uređivanje određen varijablom $VISUAL. Ako sadržaj tih varijabli nije definiran onda se pokreće editor ed za <e> odnosno vi editor za <v>.

Primjer:

Želimo ispisati sadržaj neke velike datoteke tako da možemo pregledati gotovo cijelu datoteku stranicu po stranicu. Pregledavat ćemo sistemske datoteke etc/passwd i /etc/termcap.

```
$ more /etc/passwd
```

Navedenom naredbom smo ispisali prvu stranicu datoteke. Pritiskom na tipku <enter> prelazimo na slijedeću stranicu.

Ako datoteka sadrži puno praznih redaka, opcije naredbe `more` nam pružaju mogućnost da ispišemo datoteku bez suvišnih praznih redaka. Pogledajmo prvo ispis datoteke */etc/termcap*.

```
$ more /etc/termcap
```

Ispis datoteke na ekranu se sastoji od velikog broja praznih redaka. Da bismo ih izbacili iz ispisa koristimo sljedeću naredbu:

```
$ more -s /etc/termcap
```

Važno je napomenuti da navedena naredba ne mijenja sadržaj datoteke već samo modificira ispis na zaslonu.

5.7. Naredba `less`

Naredba `less` radi gotovo istu stvar kao i naredba `more` no ima prednost da se može vraćati unatrag i pregledavati sadržaj datoteke liniju po liniju. Za pomicanje redak po redak unaprijed ili unazad mogu se koristiti tipke <pgup> i <pgdn>, za razliku od naredbe `more` kod koje je sadržaj moguće pregledavati tipkama <enter> i <space> te tipkom <b> ako se želimo vratiti jednu stranicu unatrag.

Naredba `less` se koristi na isti način kao i naredba `more`, za pregled sadržaja datoteke potrebno je

samo navesti:

```
less <imedatoteke>.
```

Naredbe `less` i `more` omogućuju pokretanje ljuske pritiskom tipke !. Nakon što se pokrene naredba !, ulazi se u podljusku koju se može napustiti naredbom za odjavu sa sustava, ^D ili exit. Program se vraća na mjesto gdje smo napustili datoteku. Ako se pritiskom tipke ^D javi pogreška onda je potrebno pokrenuti naredbu `logout` da bi izašli iz podljuske u datoteku.

5.8. Zadaci

1) Naredbama `head` i `tail` pregledati prvih 20 i zadnjih 20 redaka datoteka /*etc/passwd* i */etc/termcap*

2) Naredbom `cat` pregledati zajedno sadržaj datoteka /etc/passwd i etc/termcap i zaustaviti ispis na zaslonu.

3) Naredbom `more` pregledati sadržaj datoteke `/etc/passwd`. Sadržaj treba pregledavati stranicu po stranicu i redak po redak, preskočiti 50 redaka pri ispisu, vratiti se 2 stranice unatrag i skočiti 3 stranice unaprijed.

4) Naredbom `less` pregledavati sadržaj datoteke `/etc/termcap`, pregledati 3. stranicu ispisa i vratiti se desetak redaka unatrag, redak po redak. Izbaciti suvišan broj praznih redaka u datoteci.

5) Naredbom `file` provjeriti tipove datoteka u trenutnom direktoriju.

6) Bilo kojoj datoteci u trenutnom direktoriju promijeniti vremenske oznake na proizvoljni datum.

7) Spojiti se na računalo pinus(ako je moguće). U datoteci `/etc/passwd` pronaći redak sa imenom vama najdražeg asistenta ili profesora :).

6. Vlasništvo i dozvole

6.1. Uvod

Linux je višekorisnički sustav i kao takav mora biti u stanju raspoznati korisnike, također mora osigurati da su svi programi koje pokrećete i datoteke koje napravite vaše vlasništvo. Linux ne dopušta da čitate ili mijenjate datoteku u vlasništvu nekog drugog korisnika, osim ako vam taj korisnik ili administrator nisu dali dopuštenje.

Dozvole za rad s datotekama određuju tko može čitati, pisati ili izvršavati datoteku, one također određuju tip datoteke.

6.2. Vlasnik, grupa i ostali

Koristeći naredbu `ls -l` na ekranu dobivamo detaljan ispis datoteka u trenutnom direktoriju:

```
$ ls -l
drwx------  2  mcavka  users    512  Jan   1   13:44   Mail
drwx------  5  mcavka  users   1024  Jan  17   08:22   News
-rw-------  1  mcavka  users   1268  Dec   7   15:12   biblio
drwx------  2  mcavka  users    512  Dec  15   21:18   bin
-rw-------  1  mcavka  users  42787  Oct  20   06:59   books
drwxr-xr-x  3  mcavka  users    512  Apr  19   11:43   public_html
```

Prvi stupac prikazuje tip i dozvole za rad s datotekom, drugi prikazuje broj linkova na datoteku (ili dodatne blokove u direktoriju), treći stupac prikazuje tko je vlasnik datoteke. Četvrti stupac prikazuje grupu kojoj datoteka pripada, peti prikazuje broj bajtova u datoteci, šesti prikazuje datum i vrijeme stvaranja, ako je datoteka stvorena prije više od jedne godine, onda se prikazuje godina stvaranja. Sedmi stupac prikazuje ime datoteke.

Polje dozvola (prvi stupac) je podijeljeno u četiri podpolja:

```
-  rwx  rwx  rwx
```

Prvo podpolje (veličine samo jednog znaka) prikazuje tip datoteke, neke od mogućih vrijednosti prikazane su u tablici 6.1.

Tablica 6.1: Vrijednosti polja tip datoteke

Znak	Značenje
-	Obična datoteka
b	Blok orijentirana posebna datoteka
c	Znakovno orijentirana posebna datoteka
d	Direktorij
l	Simbolički link
p	FIFO sturktura

Sljedeća tri podpolja pokazuju dozvole za čitanje, pisanje i izvršavanje datoteke. Slova rwx u prvom od tih podpolja znače da datoteka ima dozvole čitanja, pisanja i izvršavanja samo za vlasnika. Sljedeće podpolje pokazuje iste informacije za grupu u kojoj se nalazi vlasnik datoteke. Posljednje podpolje pokazuje dozvole za sve ostale korisnike sustava.

Prilikom stvaranja datoteka, automatski joj se dodjeljuju dozvole definirane u varijabli "umask" (engl. *file creation mask*). Izvršavanjem naredbe umask dobivamo oktalan ispis trenutnog statusa varijable (možemo dobiti višak ili manjak vodećih ništica). Ako želimo promijeniti varijablu, koristimo: umask xyz (gdje su xyz oktalne znamenke 0-7). No moramo zapamtiti da su vrijednosti xyz zapravo komplementi stvarnih vrijednosti dozvola koje želimo zadati.

Ako na sustavu upišemo `umask` dobit ćemo nešto ovog oblika:

```
$ umask
0022
```

Primjer:

Želimo da se prilikom stvaranja datoteka automatski dodaju dozvole `rwx` samo za vlasnika datoteke, a za grupu i ostale samo `r`.

Prvo binarno zapisujemo kako nam dozvole trebaju izgledati: $(111\ 100\ 100)_2 = (744)_8$

Zatim radimo komplement $(000\ 011\ 011)_2 = (033)_8$

I na kaju pozivamo naredbu:

```
$ umask 033
```

6.3. Promjena dozvola

Mijenjanje dozvola izvodi vlasnik datoteke naredbom `chmod` i to na dva načina: "oktalno" i "simbolički".

Sintaksa:

```
chmod <prava_pristupa> <datoteka_na_koju_se_prava_odnose>
```

a) oktalno

Kod ovog načina, prava pristupa se prikazuju s tri oktalne znamenke (brojevi od 0-7). Prva oktalna znamenka se odnosi na vlasnika, druga znamenka se odnosi na grupu kojoj datoteka pripada, a treća na sve ostale korisnike računala. Vrijednosti oktalnih znamenki, te prava koja određuju prikazana su u tablici 6.2.

Tablica 6.2: Vrijednost oktalnih znamenki i prava koja određuju

Oktalna znamenka	Binarno	Prava
0	000	---
1	001	--x
2	010	-w-
3	011	-wx
4	100	r--
5	101	r-x
6	110	rw-
7	111	rwx

Primjer:

Stvorimo direktoriji `proba` i u njemu datoteku `unix.txt`.

Želimo datoteci `unix.txt` promjeniti dozvole tako da vlasnik ima sva prava, grupa pravo čitanja i pisanja, a ostali korisnici samo pravo čitanja.

```
$ chmod 764 unix.txt
$ ls -l
total 4
-rwxrw-r--  1 mcavka mcavka 0 Oct 27 00:15 unix.txt
```

Primjer:

Želimo zabraniti ispis sadržaja direktorija `proba.`

Oduzimamo dozvolu čitanja.

```
$ chmod 300 proba
$ ls -l
total 16
drwxr-xr-x  2 mcavka mcavka 4096 Oct 27 00:11 Desktop
d-wx------  2 mcavka mcavka 4096 Oct 27 00:15 proba
$ ls proba
ls: proba: Permission denied
```

Primjer:

Želimo zabraniti pristup direktoriju `proba`, svim datotekama i poddirektorijima koji se nalaze unutar direktrorija `proba`.

Oduzimamo dozvolu izvršavanja.

```
$ chmod 600 proba
$ ls -l
total 16
drwxr-xr-x  2 mcavka mcavka 4096 Oct 27 00:11 Desktop
drw-------  2 mcavka mcavka 4096 Oct 27 00:15 proba
$ cd proba
bash: cd: proba: Permission denied
$ cat proba/unix.txt
cat: proba/unix.txt: Permission denied
```

b) simbolički

Mora se definirati: kome dajete dozvolu, koje operacije (dodavanje, oduzimanje ili postavljane) i koje dozvole.U tablici 6.3 prikazani su parametri naredbe.

Tablica 6.3: Parametri naredbe chmod

	Vrijednost	Značenje
Tko	a	Svi korisnici (vlasnik, vlasnikova grupa, svi ostali korisnici)
	g	Vlasnikova grupa
	o	Svi ostali korisnici
	u	Samo vlasnik
Operator	+	Dodaje mod
	-	Oduzima mod
	=	Postavlja apsolutnu vrijednost moda
Dozvola	r	Postavlja dozvolu čitanja
	w	Postavlja dozvolu pisanja
	x	Postavlja dozvolu izvršavanja

Primjer:

Želimo datoteci `unix.txt` promijeniti dozvole tako da vlasniku oduzmemo pravo pisanja, a grupi oduzmemo pravo čitanja.

```
$ chmod u-w,g-r unix.txt
$ ls -l
total 4
-r-x-w-r--  1 mcavka mcavka 0 Oct 27 00:15 unix.txt
```

6.4. Promjena vlasnika

Promjenu vlasnika nad nekom datotekom radimo naredbom `chown`, ali pritom moramo paziti jer nakon promjene vlasnika datoteke, datoteka više nije u našem posjedu. Ovu naredbu može jedino administrator koristiti.

Sintaksa:

```
chown <korisnik> <ime_datoteke>
```

Primjer:

Želimo korisniku `mcavka` predati datoteku `unix.txt`.

```
$ chown mcavka unix.txt
```

6.5. Promjena grupe

Slično kao i promjenu vlasnika, vršimo i promjenu grupe, naredbom `chgrp`. Uvjet je da moramo biti "član" te grupe. Ovu naredbu također može jedino administrator koristiti.

Sintaksa:

```
chgrp <grupa> <ime_datoteke>
```

Primjer:

Želimo grupi `grupa1337` predati datoteku `unix.txt`.

```
$ chgrp grupa1337 unix.txt
```

6.6. Zadaci

1) Neka svaki korisnik kreira po jednu datoteku `obrisi_me` (pomoć: koristiti naredbu `touch`) i neka joj dodijeli dozvole `rwx` za sve druge korisnike. Zadatak je pronaći naredbom `find` datototeku `obrisi_me` nekog drugog korisnika te istu obrisati iz njegovog foldera.

2) Stvoriti datoteku, oktalno postaviti dozvole pisanja i izvršavanja za vlasnika. Probati vidjeti sadržaj datoteke. Simbolički dodati dozvolu čitanja. Ponovo probati pročitati datoteku. Što možete zaključiti?

3) Podesiti da se prilikom stvaranja novih datoteka automatski kreiraju dozvole čitanja, pisanja i izvršavanja samo za vlasnika.

7. Pretraživanja, filtri i cjevovodi

7.1. Preusmjeravanje sadržaja, operatori <, >, >>, |

Svakom procesu pokrenutom na Linux-u definiraju se datoteke standardni ulaz (stdin), standardni izlaz (stdout) i standardni izlaz za greške (stderr). U standardni ulaz se pohranjuju ulazni podaci (npr. s tipkovnice), a u standardni izlaz se pohranjuju podatci koji se ispisuju na zaslon kao rezultat neke naredbe (npr. `ls`). Linux korisniku nudi mogućnost preusmjeravanja proizvoljne datoteke na standardni ulaz, odnosno standardni izlaz. Također moguće je preusmjeriti podatke koji se šalju na standardni izlaz u neku proizvoljnu datoteku te nad njima vršiti obradu.U tu svrhu koriste se operatori preusmjeravanja `< , > i >>.`

Operatorom **>** podatci, koji bi se trebali poslati na standardni izlaz, preusmjeravaju se u neku drugu datoteku po izboru korisnika.

Primjer:

Želimo sadržaj direktorija u kojemu se nalazimo pospremiti u datoteku `test`. To možemo ostvariti korištenjem operatora > na sljedeći način:

```
$ ls -l > test
```

Ukoliko datoteka `test` ne postoji na disku, stvoriti će se nova prazna datoteka pod tim imenom. U slučaju da datoteka `test` već postoji, njen sadržaj će se prebrisati. Primjećuje se da na zaslonu nema očekivanog ispisa naredbe `ls`. Podatci koje je naredba `ls` poslala na standardni izlaz preusmjereni su i pospremljeni u datoteku `test`.

Korištenjem operatora < datoteka se povezuje sa standarnim ulazom naredbe.

Primjer:

Želimo li provjeriti što se nalazi u datoteci test, koju smo formirali u prethodnom primjeru, možemo iskoristiti sljedeće jednostavno preusmjeravanje.

```
$ cat < test
```

Na zaslonu je sada vidljiv ispis sadržaja datoteke `test`. Datoteka test se prosljeđuje na ulaz naredbe `cat` koja zatim ispisuje njen sadržaj na zaslon.

Operator >> ima sličnu funkciju kao i operator > uz razliku što nove podatke koji se šalju na standardni izlaz preusmjerava i sprema na kraj datoteke pritom ne brišući njen sadržaj.

Prilikom preusmjeravajna dešava se nakoliko uzastopnih operacija. Prvo se otvara nova datoteka. Za korisnika upoznatim sa radom u programskom jeziku C ta je operacija ekvivalentna programskom odsječku: `f = fopen ("test", "w");` (za operator >> taj dio programskog odsječka u C-u glasi: `f = fopen ("test", "a");`. Zatim se izvršava naredba te se njen izlaz upisuje u datoteku. U slučaju operatora < se iz datoteke čitaju podaci i prosljeđuju naredbi. Na kraju se datoteka zatvara: `fclose (f);`. Ukoliko naredba nije dobro zadana, preusmjeravanje se izvrši u standardni izlaz za pogreške (`stderr`).

Primjer:

Ovim primjerom želi se pokazati što se dogodi ukoliko korisnik greškom unese nepostojeću naredbu te njen izlaz pokuša preusmjeriti u datoteku.

```
$ lslj -l > test
```

Stvoriti će se nova datoteka `test` no u nju se neće ništa preusmjeriti jer je došlo do greške. Na zaslonu će se ispisati obavjest da je došlo do greške. U tom slučaju ulogu standardnog izlaza preuzima standardni izlaz za greške (`stderr`) kojemu nije moguće fizički pristupiti.

Primjer:

Postoji mogućnost međusobnog ulančavanja operatora preusmjeravanja. Želimo preusmjeriti datoteku `test` u datoteku `test2` .

```
$ cat -v < test > test2
```

Datoteka `test` se preusmjerava na ulaz naredbe `cat` čiji je standardni izlaz preusmjeren u datoteku `test2`. Ovime se stvara nova datoteka `test2` u koju je zapravo kopiran sadržaj datoteke `test1`.

Linux nudi mogućnost ulančavanja procesa stvaranjem **cjevovda** (engl. Pipeline). Cjevovodi služe za povezivanje jednog ili više programa u cjelinu.Standardni izlaz jednog programa prosljeđuje se na standardni ulaz drugog programa. Na taj način moguće je iz programa određene funkcionalnosti dobiti složeniju funkcionalnost. Povezivanje se ostvaruje na jednostavan, način pomoću simbola **|**.

Primjer:

Želimo upotrebom dvije jednostavne naredbe postići složeniju cjelinu. Koristiti ćemo naredbu `who` pomoću koje možemo saznati podatke o korisnicima spojenim na sistem i naredbu `head` kojom možemo ispisati datoteku na zaslon. Zanimaju nas podaci vezani uz samo prvih 5 korisnika.

```
$ who | head -5
```

Naredba `who` na svoj standardni izlaz šalje podatke o korisnicima koji su spojeni na sistem. Njen standardni izlaz spojen je na standardni ulaz naredbe `head` koja ispisuje samo prvih 5 radaka datoteke (opcija `-5`) primljene od naredbe `who`.

Želimo li ispisati prvih pet redaka datoteke `/etc/passwd` možemo formirati cjevovod od naredbi `cat` i `head` na sljedeći način:

```
$ cat /etc/passwd | head -5
```

Na zaslonu dobivamo prvih pet redaka datoteke `/etc/passwd`. Vrlo je korisno poznavati na koji su način podaci organizirani u datoteci i koje je njihovo značenje. Na taj način moguće je saznati željene informacije. U konkretnom slučaju datoteka `/etc/passwd` sadrži podatke o korisnicima sustava. Da bi smo lakše izdvojili podatke koji nas zanimaju često upotrebljavamo naredbe opisane u sljedećih nekoliko poglavalja.

7.2. Izdvajanje određenih redaka datoteke, naredba `grep`

Naredba `grep` je filter koji služi za traženje uzorka (engl. Pattern) u nekoj datoteci. Nakon što pronađe traženi uzorak, ispisuje liniju u kojoj se taj uzorak nalazi na zaslon. Filtri su naredbe koje se često koriste kao integralni dijelovi cjevovoda na taj način da vrše obradu nad podatcima koje

dobivaju na svoj standardni ulaz te potom obrađene podatke šalju na svoj standardni izlaz.

Naredba `grep` je korisna prilikom traženja izgubljenih datoteka ili datoteka koje sadrže tekst koji korisnik traži. Želimo li pronaći neku datoteku u direktoriju, možemo konstruirati cjevovod na taj način da izlaz naredbe `ls` prosljedimo naredbi `grep`.

Sintaksa:

```
grep [<zastavice>] <uzorak> <datoteka>
```

Zastavice naredbe grep prikazane su u tablici 7.1:

Tablica 7.1: Zastavice naredbe grep

Zastavica	Značenje
-c	spisuje koliko ima linija u datoteci koje sadržavaju zadani uzorak
-i	zanemaruje razliku između velikih i malih slova
-l	ispisuje imena datoteka koje sadrže zadani uzorak
-n	ispisuje i redni broj linije koja sadrži uzorak
-v	ispisuje linije koje ne sadržavaju zadani uzorak

Primjeri:

Unos naredbe:

```
$ ls -l > test
```

Standardni izlaz naredbe `ls` preusmjerava se u datoteku `test`. U njoj se sada nalazi sadržaj direktorija u kojemu se nalazimo.

Želimo ispisati sve linije datoteke test koje sadržavaju neki nama zanimljivi uzorak.

```
$ grep linux test
```

Ispisuju se sve linije datoteke test koje sadržavaju traženi uzorak (“linux”). Na taj način možemo doći do popisa datoteka koje u svome imenu sadržavaju traženi uzorak.

Želimo li da naš uzorak ne bude osjetljiv na velika i mala slova možemo koristiti opciju `-i`.

```
$ grep -i linux test
```

Unošenjem zastavice -i prilikom pretraživanja nareba grep neće praviti razliku između velikih i malih slova (npr. ispisati će i one linije koje sadrže uzorak LINUX ili neku drugu kombinaciju velikih i malih slova u tom uzorku što nije bio slučaj u prethodnom primjeru).

Kompleksnija pretraživanja moguće je ostvariti naredbama `egrep` i `fgrep`.

7.3. Brojanje riječi i linija datoteke, naredba `wc`

Naredba **w**ord **c**ount služi za brojanje riječi i linija unutar datoteke. Pomoću nje možemo dobiti informaciju koliko naša datoteka sadrži linija, riječi i znakova.

Sintaksa:

```
wc [<zastavica>] <datoteka>
```

Primjer:

U ovom jednostavnom primjeru želimo saznati koliko datoteka test sadrži riječi, linija i znakova.

```
$ wc /etc/passwd
4 34 123
```

Ispis je formatiran na način da prvi broj predstavlja broj linija, drugi broj riječi, a treći broj znakova. Znači da datoteka test sadrži 4 linije, 34 riječi i 123 znaka.

Ekvivalentna naredba korištenjem preusmjeravajna je:

```
$ wc < /etc/passwd
```

Zastavice naredbe wc prikazane su u tablici 7.2:

Tablica 7.2: Zastavice naredbe wc

Zastavica	Značenje
-w	broji samo riječi
-l	broji samo linije
-c	broji samo znakove

Primjeri:

Možemo koristiti neku od kombinacija zastavica da postignemo ispis željenih podataka na zaslonu.

```
$ wc -wl termcap
```

Ispisuje broj riječi i linija na zaslon. Moguće su i druge kombinacije zastavica.

Da bi smo saznali koliko je korisnika spojeno na sistem možemo koristiti naredbu:

```
$ who | wc -l
```

Prilikom ispisa naredba *who* za svakog korisnika rezervira jednu liniju. Prebrojimo li broj linija dobivamo informaciju o tome koliko je korisnika spojeno na sistem.

```
$ ls | wc
```

Korištenjem cjevovoda u ovom se slučaju preusmjerava standardni izlaz naredbe `ls` u standardni ulaz naredbe `wc`. Prebrojavamo koliko ima riječi, linija i znakova u ispisu naredbe `ls`.

7.4. Sortiranje sadržaja datoteke, naredba `sort`

Naredba `sort` jedan je od najkorisnijih filtera i često se koristi kao interni dio cjevovoda na mjestu gdje treba sortirati podatke.

Podatke koje dobiva na ulazu sortira po abecedi i šalje ih na standardni izlaz.

Sintaksa:

```
sort [<zastavice>] <datoteka>
```

Upotrebom zastavica može vršiti više različitih načina sortiranja.

Zastavice naredbe sort prikazane su u tablici 7.3:

Tablica 7.3: Zastavice naredbe sort

Zastavica	Značenje
-b	zanemaruje početne praznine
-d	sortira po abecednom redu(razlikuje velika i mala slova)
-f	sortira po abecednom redu (ne razlikuje velika i mala slova)
-n	sortira po numeričkom redosljedu
-r	sortira u obrnutom redosljedu

Primjeri:

Želimo sortirati imena datoteka u direktoriju po abecednom redu bez obzira na velika i mala slova u imenu. Konstruitrati ćemo cjevovod od naredbi `ls` i `sort`.

```
$ ls -l | sort -f
```

Naredba `ls` svoj izlaz šalje na ulaz naredbi `sort` koja podatke zatim sortira na željeni način koji smo izabrali upotrebom zastavice `-f`. Naredba `sort` ako se nalazi na kraju cjevovoda ispisuje obrađene podatke na zaslon.

Datoteke u direktoriju treba sortirati po njihovoj veličini. Koristiti ćemo opciju naredbe `ls` koja pri ispisu na prvo mjesto kao podatak ispisuje veličinu datoteke. Zatim ćemo dobivene podatke sortirati po numeričkom redosljedu pomoću naredbe `sort` koristeći njenu opciju `-n`.

```
$ ls -s | sort -n
```

Upotrebom naredaba du i sort možemo sortirati direktorije i njihove poddirektorije po veličini.

```
$ du -S -k | sort -n -r
```

Ispisani direktoriji su sortirani po veličini u Kb od najvećeg prema najmanjem zbog korištenja opcije `-i` naredbe `sort`. Veličina poddirektorija nije uračunata u veličinu njihovih matičnih direktorija zbog korištenja opcije `-S` naredbe `du`. Opcijom `-k` ostvarujemo ispis veličine u Kb umjesto u blokovima (512b).

7.5. Izbacivanje duplih linija, naredba `uniq`

Ponekad se u datoteci nalazi puno identičnih linija (praznih ili sa upisanim podatcima). Da bi se datoteke očistile od tog viška linija koristi se naredba `uniq`. Naredba zadanu datoteku obrađuje (ne mijenjajući njen sadržaj) i na standardni izlaz šalje obrađeni sadržaj datoteke (bez suvišnih linija).

Sintaksa:

```
uniq [<zastavice>] <datoteka> <datoteka2>
```

Zastavice naredbe `uniq` prikazane su u tablici 7.4:

Tablica 7.4: Zastavice naredbe uniq

Zastavice	Značenje
-u	ispisuje linije koje se ne ponavljaju
-d	ispisuje linije koje su se ponavljale
-c	ispisuje broj koliko se puta linija ponovila

Naredba se ponaša u skladu sa zastavicama koje je postavio korisnik. Pročišćenu datoteku sprema u `datoteka2`, a ako `datoteka2` nije navedena šalje je na standardni izlaz (ispisuje na zaslon).

Primjer:

Želimo ispisati linije u datoteci test koje se ne ponavljaju. Naredba `uniq` uklanja samo uzastopne linije koje se ponavljaju. Ukoliko postoje identične linije u datoteci koje se ne nalaze jedna iza druge, naredba uniq neće imati utjecaja.

```
$ uniq -u test
```

Naredba ispisuje linije koje se ne ponavljaju u datoteci `test`.

Želimo li proćišćenu datoteku spremiti u neku novu datoteku možemo to učiniti navođenjem imena nove datoteke iza originalne.

```
$ uniq test test.new
```

Standardni izlaz naredbe `uniq` sprema se u datoteku `test.`*`new`* koja sada sadrži samo jedinstvene linije iz datoteke `test`.

7.6. Izrezivanje dijelova redaka datoteka, naredba `cut`

Naredba `cut` omogućuje izrezivanje dijelova linije u datoteci. Moguće je odabrati dio datoteke koji se želi izdvojiti (zada se opseg).

Sintaksa:

```
cut [<zastavice i opseg>] <datoteka>
```

Zastavice naredbe cut prikazane su u tablici 7.5:

Tablica 7.5: Zastavice naredbe cut

Zastavice	Značenje
-c	izrezuje znakove iz opsega (stupce)
-d	određuje ograničivač (zadnji znak u polju) koji se izrezuje. Koristi se u kombinaciji za zastavicom f.
-f	izrezuje polja zadana u listi

Primjeri:

Želimo obraditi datoteku */etc/passwd* na način da iz nje izrežemo stupce 1. do 5. mjesta, 7. stupac i od 9. stupca do kraja. Dobivene podatke treba zatim sortirati i ispisati. Opseg se zadaje neposredno iza zastavice. Moguće je navesti više opsega na način da ih se odvoji zarezom. Naredba sort sortira podatke abecednim redosljedom te ih potom ispisuje na zaslon.

```
$ cut -c1-5,7,9- etc/passwd | sort
```

Želimo saznati samo imena korisnika koji su spojeni na sistem. Koristiti ćemo činjenicu da naredba `who` ,prilikom ispisa podataka, na prvo mjesto stavlja ime korisnika i odvaja ga razmakom od sljedećeg podatka. Zato ćemo za ograničivač polja (završni stupac do kojega se polje izrezuje) izabrati upravo " ". Ograničivač polja koristi se isključivo u kombinaciji sa zastavicom `-f` koja služi za odabir polja koje želimo izabrati.

```
$ who | cut -d" " -f1
```

7.7. Traženje datoteka

7.7.1. Naredba `find`

Naredba `find` pretražuje direktorij i pronalazi datoteke koje zadovoljavaju određene uvjete. Naredba počinje pretragu pozicioniranjem na početak mjesta traženja. Zatim uspoređuje parametre svih datoteka i zadane parametre te u slučaju da su oni isti izvršava operaciju zadanu u izvedbenom dijelu naradbe. Naredba pretražuje sve poddirektorije od početnog zato je korisno orgraničiti područje traženja. Kada nemamo informaciju o približnom položaju datoteke, kao početni direktorij zada se */* (root direktorij).

Sintaksa:

```
find <direktorij> <uvjet>
```

Uvjeti naredbe find prikazani su u tablici:

Tablica 7.6: Uvjeti naredbe find

Uvjet	Značenje
-name naziv datoteke	nalazi datoteke s imenom koje se zadaje u polju naziv datoteke
-type znak	traži datoteke koje su određenog tipa (f obična datoteka, b blok orijentirana datoteka, d direktorij i sl.)
-user korisnik	traži datoteke koje pripadaju željenom korisniku
-size n	pretražuje datoteku prema veličini koja je zadana sa n (u blokovima). Ukoliko je zadana s +n traži datoteke s više od n blokova, a ako je veličina zadana sa -n onda traži datoteke s manje od n blokova. Ukoliko se stavi oznaka nc onda uvjet za veličinu prelazi broj znakova.

Naredba `find` sadrži još brojne opcije koje se mogu saznati na stranicama priručnika (`$ man find`);

Moguće je zadati više uvjeta i povezati ih na sljedeće načine:

`-uvjet1 -o -uvijet2` mora biti zadovoljen barem jedan uvjet

`-uvjet1 -a -uvijet2` moraju biti oba zadovoljena

`\(izraz\)` povezuje izraze koji predstavljaju uvjete.Prvo se izvršavaju izrazi u zagradi.

Kada se pronađe tražena datoteka pomoću naredbe `find`, moguće je izvršiti naredbu nad tom datotekom. Ukoliko nije drugačije zadano, izvodi se opcija `-print`.

Da bi se izvršile željene naredbe koriste se sljedeće opcije prikazane u tablici 7.7:

Tablica 7.7: Opcije naredbe find

Opcija	Značenje
-print	ispisuje puno ime nađene datoteke
-exec naredba {} \	nakon što je nađena datoteka izvrši se naredba. Oznaka {} fizički predstavlja nađenu datoteku u sintaksi naredbe koja se treba izvesti
-ok naredba {} \	slično kao opcija **-exec** samo što je za izvođenje naredbe potrebna korisnikova potvrda

Napomena:

Oznake {} i ; često se moraju zaštititi navodnicima ili s oznakom \ zato što u suprotnom ljuska može argumente naredbi *find* protumačiti kao vlastite argumente.

Primjeri:

Naredbom `find` treba pronaći u direktoriju `/etc` datoteku `passwd`. Koristiti ćemo pretraživanje po imenu opcijom *`-name`*.

```
$ find /etc -name passwd
```

Ukoliko naredba `find` pronađe traženu datoteku, ispisati će njeno puno ime (`/etc/passwd`).

Koristan kriterij prilikom pretraživanja je tekođer i tip datoteke. Na taj način možemo pronaći datoteke tipa koji nas zanima. Treba naći sve obične datoteke u direktoriju `/tmp`.

```
$ find /tmp -type f
```

Želimo korištenjem naredbe `find` pronaći i obrisati sve datoteke u sistemu čija je veličina 0 blokova (prazne datoteke). Koristiti ćemo naredbu `rm` za brisanje datoteka.

```
$ find / -size 0 -ok rm {} \;
```

Naredba počinje potragu u root direktoriju te je nastavlja u svim poddirektorijima. Kada se pronađe datoteka koja ispunjava uvjete (prazne datoteke), nad njom se izvršava naredba `rm` (uz korisnikov pristanak). U samom zapisu zagrade *`{}`* predstavljaju argument naredbi `rm` (svaku pronađenu datoteku).

7.7.2. Naredba `whereis`

Korištenjem naredbe `whereis` može se dobiti informacija gdje se nalazi tražena datoteka (program).

Naredba `whereis` omogućuje korisniku da pronađe gdje se nalazi izvorni kod naredbe (engl. source), binarni zapis (engl. binary), stranice priručnika.

Sintaksa naredbe:

```
whereis [<zastavice>] <datoteka>
```

Zastavice naredbe whereis prikazane su u tablici 7.8:

Tablica 7.8: Zastavice naredbe whereis

Zastavica	Značenje
-b	traži binarne zapise
-m	traži priručnik
-s	traži izvorni kod
-B direktorij	ograničuje potragu za binarnim zapisom na zadani direktorij
-M direktorij	isto kao -B samo se odnosi na stranice priručnika
-S direktorij	isto kao -B samo se odnosi na izvorni kod traženog programa
-f	terminator upisa liste direktorija, iza njega slijedi ime datoteke (koristi se sa -B, -S, -M)

Iz imena datoteka naredba prvo odstarnjuje direktorij u kojemu se nalaze (engl. *Path*). To znači da naredba pronalazi programe unatoč tome što se može zadati pogrešan direktorij u kojemu se nalaze. Naredba `whereis` radi neovisno o $PATH varijabli. U $PATH se nalazi popis direktorija koji se pretražuju prilikom pokušaja izvođenja neke naredbe. Ukoliko se uspješno pronađe izvorni kod naredbe, ona se izvodi. Postoji mogućnost da $PATH varijabla nije dobro zadana i ne sadrži direktorij u kojemu se nalazi naredba koju želimo izvršiti. U tom slučaju možemo primjeniti naredbu `whereis` i pronaći izvorni kod naredbe neovisno o $PATH varijabli.

Primjer:

Kada pokrećemo neku naredbu, ona se nužno ne mora nalaziti u našem direktoriju. Da bi smo saznali gdje se nalazi naredba možemo koristiti naredbu `whereis`.

```
$ whereis ls
```

Upisom naredbe dobivamo informaciju o tome gdje se uistinu nalazi *`ls`*.

7.7.3. Naredba `locate`

Naredba `locate` traži i ispisuje datoteke koje u svome imenu sadrže zadani uzorak. Datoteke traži u internoj bazi podataka koju periodički osvježava program `updatedb`. Naredbu `locate` preporučeno je koristiti umjesto naredbe `find` jer ona pri pretraživanju koristi svoju internu bazu podataka pa je samo pretraživanje brže. Do problema može doći ako baza podataka nije ažurirana i ne sadržava popis novijih datoteka. Tada naredba `locate` neće dati rezultata pa korisnik može upotrjebiti naredbu `find`.

Sintaksa:

```
locate [<zastavice>] <uzorak>
```

Zastavice naredbe prikazane su u tablici 7.9:

Tablica 7.9: Zastavice naredbe locate

Zastavica	Značenje
-d direktorij	traži datoteke u zadanom direktoriju

Primjeri:

Treba pronaći datoteku `/etc/passwd` pomoću naredbe `locate`. Kao direktorij potrage izabrati `/etc`.

```
$ locate -d /etc passwd
```

U ovom slučaju naredba `locate` pronalazi traženu datoteku. Pokušajmo pronaći novu datoteku. U ovom slučaju naredba ne može dati rezultat jer baza podataka još nije ažurirana.

```
$ locate test
```

7.8. Zadaci

1) Ispisati imena prvih 5 datoteka u trenutačnom direktoriju tako da budu sortirana po numeričkom redosljedu. (napomena: realizirati naredbu upotrebom cjevovoda izgrađenog od `ls`, `sort` i `head` naredbi)

2) Stvoriti datoteku koja sadrži linije koje se ponavljaju i jedinstvene linije. Prebrojati koliko ima riječi u linijama koje se ponavljaju. Ispisati linije koje se ne ponavljaju abecednim redosljedom.(napomena: koristiti naredbe `uniq`, `sort`, `wc`)

3) Pronaći sve datoteke u direktoriju koje su veće od 50 blokova i ispisati koliko imaju linija. (napomena: koristiti naredbe `find` i `wc`)

4) Pronaći sve datoteke koje su manje od 1000 blokova i ne pripadaju niti jednom korisniku te im ime ispisati na zaslon.

5) Pronaći u kojemu se direktoriju nalazi izvorni kod naredbe uniq.

6) Stvoriti datoteku i u nju preusmjeriti izlaz naredbe `ls`. Sve linije koje sadrže određeni uzorak (zadati si ga po volji) sortirati abecedno obrnutim redosljedom te ispisati na zaslon. (napomena: koristi naredbe `grep` i `sort`)

8. Zamjenski znakovi i regularni izrazi

8.1. Zamjenski znakovi

Zamjenski znakovi su specijalni znakovi koji zamjenjuju jedan znak ili više znakova. Njihovu upotrebu odnosno zamjenu znakova omogućuje ljuska (engl. shell). Postoji nekoliko grupa znakova definiranih u standardnoj ljusci: znakovi koji zamjenjuju jedan znak, znakovi koji zamjenjuju više znakova te definirani skup znakova koji zamjenjuje neki znak. Navedene vrste zamjenskih znakova su: `*, ?` te `[.......].` Značenje navedenih znakova dano je u tablici 8.1

Tablica 8.1. Popis zamjenskih znakova

Znak	Značenje
*	mijenja više znakova
?	mijenja jedan znak
[...]	unutar uglatih zagrada odredi se interval znakova iz kojeg bilo koji od ponuđenih znakova zamjenjuje jedan znak

Primjer:

```
[m-o] zamjenjuje mala slova u intervalu od slova m do slova o
[B-E] zamjenjuje velika slova u intervalu od slova B do slova E
[A-z] zamjenjuje sva velika i mala slova od slova a do slova z
[3-7] zamjenjuje brojeve od broja 3 do broja 7
```

Postavlja se sljedeći problem: kako naći datoteke koje u nazivu sadrže neke specijalne znakove koje ljuska koristi? Naime, ljuska neke definirane znakove tumači kao naredbe za izvođenje radnji s datotekama. Ukoliko se neki od tih znakova navede unutar izraza ljuska će ga protumačiti kao instrukciju. Rezultat takve pretrage neće biti korektan te je potrebno "isključiti" (engl. escape) funkciju koju specijalan znak po svojoj definiciji obavlja. Kad se "isključi" funkcija znaka, ljuska ga više ne interpretira kao specijalan nego kao običan znak kojeg se traži ili u nazivu datoteka ili u linijama jedne odnosno više datoteka. Kako se postiže isključivanje funkcije znaka? Naravno, znakovima: znakom \ te jednostrukim i dvostrukim navodnicima.

Tablica 8.2. Isključivanje funkcije posebnih znakova

Znak	Djelovanje
\	isključi funkciju jednog posebnog znaka koji slijedi
'...'	isključi funkciju svih posebnih znakova koji su sastavni dio niza znakova unutar navodnika
"..."	isključi funkciju svih posebnih znakova, koji su sastavni dio niza znakova unutar navodnika, osim `$ ` ` ' \`. Ova četiri posebna znaka zadrže funkciju koju obavljaju

Primjer:

pronalazak i ispis svih datoteka u direktoriju koje u nazivu imaju ekstenziju `.doc`

```
$ ls *.doc
norvegian wood.doc velvet_goldmine.doc 7.doc
$
```

Primjer: pronalazak i ispis svih datoteka u direktoriju koje u svom nazivu sadrže jednu ili više znamenki

```
$ ls *0* *1* *2* *3* *4* *5* *6* *7* *8* *9*
098.doc 14cups.sxw backup0.file ruf_an.666 36abc.3
098.doc 14cups.sxw 098.doc 36abc.3
$
```

Prikazani ispis rezultata pretrage nije optimalan. Naime, neke su datoteke višestruko ispisane zbog formulacije regularnog izraza. Stoga se treba poslužiti regularnim izrazom koji će specificirati ispis svake datoteke sa znamenkom u nazivu samo jedanput:

```
$ ls *[0123456789]*
098.doc 14cups.sxw backup0.file ruf_an.666 36abc.3
$
```

Ili još kraće:

```
$ ls *[0-9]*
098.doc 14cups.sxw backup0.file ruf_an.666 36abc.3
$
```

Primjer: popis datoteka čiji nazivi sadrže brojeve samo unutar naziva

```
$ ls *[a-z][0-9][a-z]*
backup0.file
$
```

Primjer: popis datoteka čiji nazivi sadrže brojeve na oba kraja naziva

```
$ ls [0-9]*[a-z]*[0-9]
36abc.3
$
```

8.2. Regularni izrazi

8.2.1. Općenito o regularnim izrazima

Regularni izrazi su nizovi znakova koji omogućavaju da se na jednostavan način točno odredi veći skup znakova. Iako sliče zamjenskim znakovima, regularni izrazi su potpuno drugačiji. Tehnika

kreiranja vlastitih regularnih izraza je vještina koju je potrebno precizno koristiti. Naime, regularni izrazi služe i za niz drugih radnji (ovisno s kojim naredbama ih se kombinira u naredbenom retku) osim za pretraživanje. Tako se može regularnim izrazom kombiniranim s naredbom ili alatom odrediti brisanje ili pomicanje u drugi direktorij niza datoteka koje u nazivu sadrže uzorak znakova. Ukoliko se regularni izraza kreira neprecizno odnosno netočno mogu se dobiti neželjeni rezultati. Izmjena i obrada teksta često se efikasno provode upotrebom regularnih izraza. Broj regularnih izraza koji se mogu kreirati korištenjem samo malog podskupa znakova je beskonačan. Dalje u tekstu je naveden popis nekih od znakova kojima se može koristiti. Ti znakovi se navode unutar uglatih zagrada.

Tablica 8.3. Popis znakova za kreiranje regularnih izraza

Znak	Značenje operatora
c	znak c
*	nula ili više pojavljivanja znaka iza kojega se nalazi (npr `a*` znači nula ili više pojavljivanja znaka `a`)
+	jedno ili više pojavljivanje znaka iza kojeg se nalazi
?	jedno ili niti jedno pojavljivanje znaka iza kojeg se nalazi
.	zamjenjuje jedan znak (bilo koji)
[xxx] ili [x-y]	odgovara jednom znaku u nizu ili intervalu znakova koji se navode
\x	služi za "isključenje" originalnog značenja, odnosno omogućuje doslovno navođenje znaka koji inače služi kao operator regularnog izraza
^uzorak_znakova	odgovara uzorku znakova koji se nalazi na početku linije u datoteci
$uzorak_znakova	odgovara uzorku znakova koji se nalazi na kraju linije u datoteci

8.3. Pretraživanje unutar datoteka pomoću `grep` naredbi

Ovo potpoglavlje obrađuje `grep` naredbe (naziv potječe iz "global/reg.expression/printer") i način korištenja naredbi: `grep`, `egrep` i `fgrep`. Da bi se uspješno savladalo učenje te korištenje grep naredbi nužno je znati što su i kako funkcioniraju tehnike stvaranja regularnih izraza te zamjenski znakovi opisani u prethodnom poglavlju. Grep naredbom pretražuje se svaka linija datoteke. Dodatan "bonus" je da se istovremeno može obavljati pretraga linija više datoteka. Dakle, može se pretraživati jedna odnosno više datoteka. Detalji o određivanju pretrage više datoteka su navedeni kasnije u tekstu. Značenja pojedinih grep naredbi su:

Tablica 8.4. Popis naredbi

Naredba	Sintaksa
grep	traži navedeni uzorak (mogu se koristiti zamjenski znakovi)
egrep	koristi regularne izraze u uzorcima za pretragu
fgrep	koristi fiksne nizove znakova (stringove)

Naredba `fgrep` ne koristi regularne izraze niti zamjenske znakove nego konkretne nizove znakova, što znači da upotreba zamjenskih znakova u naredbenom retku uz `fgrep` nalazi doslovno identične uzorke u samoj datoteci. U naredbenom retku se dodaju odrednice (specifikacije) koje određuju način na koji će se pronađene linije datoteke (odnosno više datoteka) ispisati. Te odrednice tj. opcije se

dodaju po potrebi, grep naredbe svoju potpunu funkciju obavljaju i bez njih. Navedene su neke najčešće korištene opcije:

Tablica 8.5. Popis opcija

Opcija	Funkcija
-v	ispis linija koje se ne podudaraju s uzorkom
-c	ispis broja pronađenih linija
-i	ispis svih linija koje se podudaraju s uzorkom bez obzira na velika i mala slova
-n	ispis linija uz označavanje istih pripadnim brojem

Primjer:

```
$ grep -n friend guide.txt
1333: large extent by the window manager. This friendly program is
1999: Copy linux from a friend who may already have the software,
5666: (unfortunately, the system was being unfriendly.)
$
```

Kao što se vidi, uzorak je pronađen u linijama 1333, 1999 i 5666. Još jedna karakteristika ovih programa je ta da se ne mora svaki put iznova pisati uzorak kad god ga se želi pronaći u datoteci. Jednostavan primjer: naći više riječi u raznim datotekama. Sve što se treba napraviti jest upisati te riječi u novu datoteku kojom će se grep poslužiti.

```
$ cat mywords
wonderful
typewriter
war
$
```

Sada se obavezno poslužiti opcijom specificiranja datoteke `-f`

```
$ grep -nf mywords guide.txt
574: typewriter used to represent screen inetaction,as in
617: software since the original space war, or , more recently, emacs.
It
1998: now you must be convinced of how wonderful linux is, and alll of
the
2549: inanimate object is a wonderful way to relieve the occasional
stress
3790: warning: linux cannot currently use 33090 sectors of this
7780: to wear the magic hat it is not needed, despite the wonderful
10091: wonderful programs and configurations are available with a bit of
work
$
```

Primjetiti da je korištena opcija numeriranja linja. Obavezno pripaziti na redoslijed kojim se navode željene opcije u `grep` naredbi. Naime, `-n` mora biti naveden prije `-f`! Kada bi opcije bile napisane

suprotnim redoslijedom grep bi javio grešku jer ne bi mogao pronaći datoteku naziva n. Nekoliko jednostavnih primjera slijedi dalje u tekstu. Primjer: ispis svih linija u datoteci koje sadrže “floor61” ili “floor67”

```
$ grep floor6[17]
```

Primjer:

Ispis svih linija u datoteci koje sadrže “floor6” nakon čega ne slijede brojevi 1 ili 7

```
$ grep -v floor6[^17]
```

Primjer:

Pronalazak niza znakova koji glasi “Windows'98” ili “Windows’95”

```
$ grep “Windows'9[58]”
```

Zašto su dodani dvostruki navodnici? Naime, moraju se navesti da se znak ' ne bi interpretirao kao početak niza znakova. Navodnici isključuju funkciju specijalnih znakova kao što je ' iz naredbenog retka kako je opisano u poglavlju “Zamjenski znakovi”. Primjer: naći linije u datoteci buiding.txt koje počinju s znamenkom

```
$ grep ^[0-9] building.txt
1 basement
2 garage
3 lift
4.entrance
$
```

```
$ fgrep ^[0-9] building.txt
$
```

Naredbeni redak neće dati rezultat pretrage kao grep u prošla dva primjera zbog činjenice da `fgrep` ne može koristiti zamjenske znakove. Mora se navesti fiksan niz znakova.

Primjer:

```
$ fgrep .entrance building.txt
4.entrance
$
```

Da se koristila naredba `grep` ili `egrep`, znak `.` bi se tumačio kao zamjenski znak ili regularni izraz.

Navedimo sada neke primjere upotrebe naredbe `egrep` koja se služi regularnim izrazima. Neka je u

datoteci `knjiga` sljedeći tekst:

```
affo
alfare
saltooooooo
oodgooood
cd
```

Želimo li ispisati sve redove koji sadrže slovo `a` koje slijedi jedan znak, a nakon toga slovo `f` upisujemo:

```
$ egrep a.f knjiga
affo
alfare
$
```

Želimo li ispisati broj redova koji sadrže više od jednog pojavljivanja slova `o` upisujemo:

```
$ egrep -c oo+ knjiga
2
$
```

U složenijim regularnim izrazima mogu se koristiti i zagrade, ali u tom slučaju treba izraz navesti u dvostrukim navodnicima kako ljuska ne bi krivo interpretirala naredbu.

```
$ egrep "(o+d)+g" knjiga
oododgooood
```

U primjeru se traže redovi koji zadovoljavaju sljedeće svojstvo: jedan ili više pojavljivanja znaka `o` kojeg slijedi znak `d` je niz koji se pojavljuje jedan ili više puta prije znaka `g`.

9. Rad s procesima

9.1. Što su procesi?

Kad pregledavate man stranicu, pretražujete datoteke sa naredbom `ls`, pokrenete `vi` editor, ili pokrećete bilo koju komandu u Unixu, pokrećete jedan ili više procesa. U Unixu, bilo koji program koji se izvršava naziva se proces. Istovremeno možete imati više pokrenutih procesa. Npr. `ls -l | sort | more` poziva tri procesa: `ls`, `sort` i `more`. Procesi mogu imati različita stanja, aktivan (running) je najčešće stanje. Proces možete zaustaviti tako da pritisnete `^Z`.

9.2. Ispis procesa na sustavu, naredba `ps`

Da bi ispisali status procesa, koristite naredbu `ps` (dolazi od 'processor status'). Ako naredbi ne dodate nikakve argumente, pokazuje aktivne i zaustavljene programe samo na vašem terminalu.

Najčešće korištene zastavice za naredbu ps prikazane su u tablici 4.1.

Tablica 9.1: Zastavice naredbe ps

Zastavica	Značenje
-a	Pokazuje sve procese terminala koji su priključeni na sustav
-g	Sortira procese po sjednicama ili grupama
-u	Radi korisničko – orijentirani ispis
-w	Koristi široki format ispisa. U slučaju ponavljanja (-ww) ispisat će se koliko god je moguće od svake komande.
-x	Pokazuje sve procese na sustavu

Zastavice `-a`, `-g`, i `-x` utječu na to koliko će se informacija prikazati na ekranu. Da bi koristili `-g` ili `-x` zastavice, morate koristiti i `-a` zastavicu.

Korištenjem naredbe `ps` ispisuje se i status procesa. Značenje statusa prikazano je u tablici 9.2.

Tablica 9.2: Značenje statusa procesa

Vrijednost	Značenje
R	aktivni (running) proces
S	spavajući (sleeping) proces (20 sekundi ili manje)
I	besposlen (idle) proces (spava više od 20 sekundi)
T	zaustavljen (stopped) proces
Z	zombi proces (proces koji je završio, ali nije oslobodio resurse)

Primjer:

Želimo ispisati sve procese, te procese za trenutnog korisnika.

```
$ ps x
  PID TTY      STAT   TIME COMMAND
 3493 ?        Ss     0:00 /usr/bin/gnome-session
 3518 ?        Ss     0:00 /usr/bin/ssh-agent -s
 3552 ?        S      0:01 /usr/libexec/gconfd-2 5
 3556 ?        S      0:00 /usr/bin/gnome-keyring-daemon
 3581 ?        S      0:00 xscreensaver -nosplash
 3608 ?        Ss     0:01 /usr/bin/metacity –sm-client-id=default1
 3612 ?        Ss     0:01 gnome-panel --sm-client-id default2
 3614 ?        Ssl    0:02 nautilus --no-default-window --sm-client-id
default3
 3616 ?        Ss     0:00 gnome-volume-manager --sm-client-id default6
 3620 ?        Ss     0:00 eggcups --sm-client-id default5
 3622 ?        Ss     0:00 pam-panel-icon --sm-client-id default0
 3722 ?        S      0:00 gnome-pty-helper
 3723 pts/1    Ss     0:00 bash
 3742 pts/1    R+     0:00 ps x
```

ispisuju se svi procesi (neki procesi su izbačeni iz ispisa zbog nepreglednosti)

```
$ ps u
USER       PID %CPU %MEM    VSZ  RSS TTY      STAT START   TIME COMMAND
mtosic    3723  0.0  0.2   4452 1404 pts/1    Ss   01:29   0:00 bash
mtosic    3746  1.0  0.4   9892 2340 pts/1    T    01:32   0:00 vim
mtosic    3747  0.0  0.1   4192  768 pts/1    R+   01:32   0:00 ps u

[1]+  Stopped                 vim
```

ispisuje procese za trenutnog korisnika

9.3. Prekidanje procesa, naredba `kill`

Vrlo korisno je znati kako “ubiti” proces. U Unixu to radi naredba `kill`. Kako bi bolje razumjeli naredbu `kill`, prvo ćemo objasniti što je to signal. Signali su razne poruke koje se šalju procesu da bi ga obavijestili o raznim 'važnim' događajima. Kada se signal pošalje procesu, on se prekida u onome što trenutno obavlja te se zahtijeva obrada signala. Svaki signal ima cijeli broj koji ga predstavlja (1, 2, itd.). Postoji mnogo signala koje `kill` naredba može poslati procesu. Dodavajući zastavicu `-l` naredbi `kill` možete vidjeti koji su vam sve procesi dostupni.

Najčešće korišteni signali u naredbi `kill` prikzani su u tablici 9.3.

Tablica 9.3: Signali naredbe kill

Broj	Ime	Značenje
1	SIGHUP	Završiti s radom (Hang up)
2	SIGINT	Prekidanje (Interrupt)

Broj	Ime	Značenje
9	SIGKILL	Kill (ne može biti ignoriran)
15	SIGTERM	Softverska terminacija

Postoji više od 30 signala u Unixu. SIGHUP signal je ono što je poslano svakom pokrenutom procesu neposredno nakon što se odjavite sa sustava. SIGINT je signal koji se šalje kada pritisnete `^C`; mnogi programi reagiraju na specifične načine kada je ovaj signal primljen. SIGKILL je „Terminator“ - šalje se informacija jezgri sustava sa zahtjevom trenutnog gašenja željenog procesa; programi ga ne mogu ignorirati. Proces se odmah prekida, bez prilike da počisti iza sebe. SIGTERM je zahvalniji: zahtijeva trenutno prekidanje programa, ali mu dopušta da ukloni privremene datoteke koje je mogao stvoriti. Podrazumijevano, `kill` šalje SIGTERM odabranom procesu. Možete odabrati i druge signale, koristeći broj ili ime signala (ako koristite ime signala, koristite bez prefiksa –SIG).

Neki procesi su „žilaviji“ i mogu odoljeti slabijim signalima poput SIGTERM-a i SIGHUP-a. (U Unixu, ovo se zove prihvat (catching) signala.) Tada treba koristiti SIGKILL. Preporuča se da se SIGKILL koristi tek nakon što ste probali prekinuti proces sa SIGTERM-om.

Primjer:

Pokrenut ćemo vi editor u pozadini te ga prekinuti SIGKILL signalom.

```
$ vi &
[1] 3750
```

pokrećemo `vi` u pozadini

```
$ ps u
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
mtosic    3723  0.0  0.2   4500 1448 pts/1    Ss   01:29   0:00 bash
mtosic    3750  0.0  0.4 10276 2336 pts/1    T    01:33   0:00 vim
mtosic    3751  0.0  0.1   2440  772 pts/1    R+   01:34   0:00 ps u

[1]+  Stopped                 vim
$ kill -9 3750
```

šaljemo KILL signal vi editoru pomocu njegovog PID-a

```
$ ps u
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
mtosic    3723  0.0  0.2   4500 1448 pts/1    Ss   01:29   0:00 bash
mtosic    3752  0.0  0.1   3376  768 pts/1    R+   01:35   0:00 ps u
[1]+  Ubijeno                 vim
```

9.4. Odvijanje procesa u pozadini

9.4.1. Naredba bg

Ako se proces nastavlja bez ikakvog ispisa na ekranu i bez ikakvih zahtjeva za unosom, možete

koristiti naredbu `bg` da ga stavite u pozadinu, gdje će se odvijati dok se ne završi. Ako proces radi u aktivnom načinu (npr. sa `wget` programom skidamo neku datoteku s interneta) ljuska čeka da taj proces završi. Potrebno je napomenuti da naredbu `bg` možete koristiti tek nakon suspendiranja procesa naredbom `^Z`. Ako program ispisuje nešto na ekran ili treba unos sa tipkovnice, sistem će automatski zaustaviti njegovo izvršavanje i obavijestiti vas. Tada možete koristiti naredbu `fg` da se program dovede u aktivan način rada i nastavi sa njegovim izvršavanjem.

9.4.2. Naredba `fg`

Kada je proces zaustavljen, možete ga ponovno staviti u aktivan način rada naredbom `fg` (fg dolazi od foreground).

Puno naredbi koje rade s procesima prihvaćaju `identifikator/oznaku posla` (jobID) kao argument. Najčešće korišteni argumenti naredbe `fg` prikazani su u tablici 9.4.

Tablica 9.4: Argumenti naredbe fg

Identifikator	Značenje
%n	Redni broj procesa n
%s	Proces koji počinje slovom s
%?s	Proces koji sadrži slovo s
%%	Trenutni proces
%	Trenutni proces (isto kao i prethodno)
%+	Trenutni proces (isto kao i prethodno)
%-	Prethodni proces

Koristeći identifikator (jobID) procesa naredbom `fg`, gdje je ID procesa argument, možete pokrenuti onaj program s kojim želite raditi. Ako ne upišete ID procesa, u aktivan način rada će doći proces koji je bio zadnji zaustavljen.

Primjer:

Pokrenut ćemo vi editor u pozadini te ćemo ga dovesti u aktivan način rada

```
$ vi &
[1] 3769
```

pokrećemo `vi` u pozadini

```
$ fg
```

otvaramo `vi` editor (jer je on zadnji koji je bio pokrenut u pozadini) isto bi postigli i slijedećom naredbom

```
$ fg 1
```

jer je '1' redni broj procesa (pogledaj par redova gore i uoči redak ' [1] 3769')

9.4.3. Operator &

Program možete pokrenuti u pozadini, dopuštajući Unixu da upravlja njime. Ako program zahtijeva unos ili ispis, zaustavlja se, poput procesa koji ste stavili u pozadinu naredbom `bg` nakon što je pokrenut.

Da bi se program automatski pokrenuo u pozadini, jednostavno napišite & na kraju komandne linije.

Primjer:

Želimo pokrenuti `vi` u pozadini.

```
$ vi &
[1] 3769
```

9.5. Odvijanje procesa nakon isključenja terminala, naredba nohup

Dolazi od „No hangup signals“. Pokreće naredbu bez hangup-a. Daje instrukciju sustavu da ne prekine proces nakon što se odjavite s terminala. Nohup ne stavlja automatski program u pozadinu, to se mora napraviti eksplicitno, tako da se doda & na kraju retka. Ono što bi se inače ispisivalo na terminalu, ispisuje se u datoteku `nohup.out`

Primjer:

Želimo naredbom `wget` (koja nam služi za 'skidanje' datoteka s interneta) skinuti neku veliku datoteku. Kako bi to moglo potrajati, koristimo naredbu nohup kako bi `wget` nastavio skidanje i nakon odjavljivanja sa terminala.

```
$ nohup wget –-quiet http://www.primjer.com/neka_velika_datoteka &
```

Zastavicu `quiet` koristimo kako bi spriječili `wget` u ispisivanju poruka na terminal.

9.6. Zadaci

1) Ispiši sve procese na sustavu

2) Isprobajte razne zastavice za naredbu `ps`

3) Pokrenuti proces po želji, saznati njegov PID te ga zaustaviti SIGKILL signalom

4) Pokrenuti dva ili više procesa u pozadini, te pomoću rednog broja procesa staviti željeni proces u aktivan način rada. Staviti taj proces opet u pozadinu te u aktivan način rada staviti drugi proces.

10. Ljuske

10.1. Što je ljuska?

Ljuska je korisnički program koji se ponaša kao posrednik između korisnika i jezgre operacijskog sustava. Njena je uloga prevođenje naredbi koje korisnik unosi u terminal. Razlikuje se od ostalih programa po tome što je specijalitzirana za pozivanje drugih programa. Ljuska ima osnovne tri upotrebe: interaktivno korištenje, podešavanje okruženja i programiranje. Interaktivno korištenje pritom označava rad u komandnoj liniji takav da ljuska čeka unos jedne ili više naredbi i tipku ENTER da ih izvrši naredbu po naredbu. Podešavanje okruženja označava podešavanje varijabli okruženja koje postoje u Unix-u, a definiraju korisnički direktorij itd. Neke varijable su već unaprijed definirane u sustavu, ali neke od njih i mi možemo podesiti u inicijalnim konfiguracijskim datotekama koje se pokreću po korisnikovoj uspješnoj autentifikaciji i autorizaciji. Programiranje u ljusci vrši se spajanjem više naredbe u jednu, izvršnu datoteku, tzv shell script

10.2. Vrste ljuski i razlike među njima

Budući da ljuska nije dio jezgre sustava već korisnički program, svatko može napisati svoju ljusku. Pritom svaki korisnik ima svoju kopiju ljuske pa prema tome ljuska može biti različita za svakog korisnika. Međutim, bilo bi poželjno da bude rasprostranjena na svim Unix-ima kako bi se postigla kompatibilnost na različitim računalima. Od mnogo inačica ljuski tri najpoznatije su: bash (***B**ourne **a**gain **s**hell* nastala od `sh` - *bourne shel*l)), `zsh` i `tcsh` (*turbo c-shell*).

10.2.1. Bash ljuska

Bash ljuska je novija inačica Bourne ljuske (*sh)*. Osim za interpretiranje naredbi korisnika, koristi se i za pisanje programskih skripti (shell scripts*)*. Skripte se pišu u bilo kojem programu za rad sa datotekama (`vi`). Mogučnost pisanja skripti je posljedica načina rada ljuske, odnosno posljedica interpretiranja naredbi koje zadaje korisnik. Svejedno je da li je unos zadan kao više pojedinačnih naredbi, ili lista (datoteka) naredbi. Bash ljuska ima mogućnost pamčenja unosa naredbi. Strelicama na tipkovnici (gore, dolje) korisnik može pregledavati prije unešene naredbe. Lista unešenih naredbi ispisuje se naredbom `history.`

10.3. Identifikacija ljuske

Postoji mnogo načina identifikacije ljuske, ali najjednostavniji način je otvoriti datoteku `/etc/passwd` i potražiti podatak uz username korisnika naredbom `grep <username> /etc/passwd`.

```
$ grep <username> /etc/passwd
username::0:0:group::/bin/bash
```

Iz niza razloga (normalno funkcioniranje vecine raznoraznih Unixoidnih programa) datoteka `/etc/passwd` mora uvijek biti citljiva svim korisnicima. Jedna jednostavna tehnika identifikacije ljuske je provjeriti što piše u komandnoj liniji. Ako komandna linija sadrži %, najvjerojatnije je riječ o tcsh ljuski. Ako komandna linija sadrži `$`, tada se najvjerojatnije koristi Bourne again (`bash`) ljuska. Identifikacija se može provesti pregledom trenutno korištenih programa u operacijskom sustavu naredbom `ps.` U koloni CMD se nalazi ime korištene ljuske. (u primjeru je to bash)

```
$ ps
  PID TTY          TIME CMD
 3353 pts/0    00:00:00 bash
 3413 pts/0    00:00:00 ps
```

Još jedan način identifikacije ljuske jest naredba `env`. Naredba ispisuje varijable okruženja, od kojih je jedna `SHELL` (ljuska koja se koristi pri registraciji korisnika). To se najjednostavnije postiže korištenjem `grep` naredbe na sljedeći način:

```
$ env|grep SHELL
SHELL=/bin/bash
```

Ljuska se može identificirati još i pomoću `chsh` naredbe, koja je opisana u nastavku.

10.4. Promjena ljuske, naredba `chsh`

Ljuska se može mijenjati samo ako je to na sustavu dozvoljeno. U slučaju da korisnik želi promijeniti ljusku, može to lako napraviti pomoću naredbe `chsh` (*change shell)*. Naredba se može koristiti bez obzira u kojem se direktoriju korisnik nalazi. Sve dozvoljene ljuske **moraju** se nalaziti u datoteci `/etc/shells`. Upisivanjem naredbe `cat /etc/shells` na ekranu se ispisuje lista dostupnih ljuski:

```
$ cat /etc/shells
/bin/sh
/bin/bash
/sbin/nologin
/bin/ash
/bin/bsh
/bin/ksh
/usr/bin/ksh
/usr/bin/pdksh
/bin/tcsh
/bin/csh
```

Kada su korisniku poznate ljuske na raspolaganju, može pristupiti promjeni ljuske. To se radi već spomenutom naredbom `chsh`. Utipkavanjem chsh, sustav će zatražiti korisnikovu lozinku, utipkavanjem lozinke, na ekran se ispisuje *glavna ljuska (login shell)*, te se traži upisivanje nove glavne ljuske:

```
$ chsh
Password:
Changing the login shell for username
Enter the new value, or press return for the default
        Login Shell [/bin/bash]:_
```

Korisnik može sada upisati jednu od ljuski na raspolaganju. U slučaju da korisnik želi biti siguran da je došlo do promjene, valja upisati: `echo $SHELL` (prema konvenciji, sve varijable ljuski se moraju pisati velikim slovima). Nakon toga sustav će ispisati ime ljuske koja je trenutno u uporabi. Zbog toga što je ljuska već otprije pokrenuta, promjena će nastupiti tek ponovnim ulogiravanjem na terminal, ili

otvaranjem novog terminala.

10.5. Okruženje i varijable, naredba `export`

Ljuska sadrži dva skupa varijabli, *lokalne varijable,* i *varijable okruženja* (*environment variables*). Njihov odnos je sličan lokalnim i globalnim varijablama u programskim jezicima tipa C. *Lokalne varijable* su poznate samo trenutnoj ljusci. *Varijable okruženja* ljuske su poznate ljusci i prenose se svim procesima koje je započela trenutna ljuska.

Ispisivanje varijabli na zaslon vrši se naredbom echo. Naredba ispisuje ono što joj se preda u listu argumenata, ako je to varijabla, potrebno je staviti znak '$' ispred. Varijabla se uklanja naredbom unset. Varijable u ljusci se postavljaju operatorom pridruživanja '=', i definirane su samo u procesu u kojemu su nastale. U sljedećem primjeru je upisana vrijednost 1 u novu varijablu A, a zatim je ispisana vrijednost te varijable te je zatim pokrenut novi proces ljuske bash. Unutar te nove ljuske nema varijable A, jer varijabla nije objavljena, odnosno nije globalna.

```
$ A=1
$ echo $A
1
$ bash
$ echo $A
$ exit
exit
$ echo $A
1
```

U slučaju da se želi koristiti vrijednost neke varijable u kombinaciji sa znakovima, npr. prikazati vrijednost varijable A, i odmah iza te vrijednosti slovo B, koriste se vitičaste zagrade '{,}', primjerice:

```
$ A=1
$ echo $AB

$ echo ${A}B
1B
```

`export` je interna naredba ljuske koja prebacuje *lokalne varijable* u *varijable okruženja* .

Sintaksa funkcije `export`:

```
export <varijabla> ili export <varijabla>=<vrijednost>
```

Argumenti funkcije su varijable koje želite staviti u *okruženje ljuske* (*shell environment*). Može se korisiti `export` čak i ako varijabla nema vrijednost. Kada joj se dodijeli vrijednost, automatski će se prebaciti u *okruženje*. Objavljene varijable nalaze se u okruženju ljuske. Kada ljuska pokrene novi proces–dijete (podproces) predaje okruženje novom procesu. Okruženje se sastoji samo od varijabli koje su objavljene. Korištenjem objavljenih varijabli, može se proslijediti informacija bilo kojem podrprocesu. Naravno, ne mora svaki podproces koristiti okruženje. Isto tako podproces ne može promijeniti okruženje roditeljskog procesa, tj. može mijenjati samo svoje okruženje. Ako se u promptu upiše `export` bez argumenata, izlistat će se objavljene varijable (za listu svih varijabli koje

su predane podljusci koristi se `env` naredbu). Za ilustraciju, navedimo primjer. Vi editor koristi `SHELL` i `TERM` globalne varijable (varijable okruženja) za definiranje terminala na kojem radi, i ljuske kojoj šalje zahtjeve. Ako korisnik želi promijeniti varijablu `EDITOR`, i staviti je u okruženje ljuske, treba upisati:

```
$ echo $EDITOR
vi
$ unset EDITOR
$ echo $EDITOR
$ export EDITOR=vim
$ echo $EDITOR
vim
```

Koristeći naredbu *`grep`*, može se provjeriti da li je varijabla objavljena.

```
$ export | grep EDITOR
declare -x EDITOR="vim"
```

Sada se može vidjeti da je `EDITOR` varijabla u okruženju ljuske, i postavljena je na vrijednost `vim`.

10.5.1. Varijable okruženja

Utipkavajnem naredbe `env` dobije se lista varijabli. Te varijable se nazivaju *varijable okruženja*, ili *environment varijable.* Kada se korisnik prijavi na sustav, environment varijable definiraju većinu stvari koja je potrebna sustavu za korištenje. Npr. matični direktorij, ljusku koja se koristi pri prijavi, ime korisnika itd. Popis bitnijih varijabli okruženja naveden je u tablici 10.1.

Tablica 10.1. Popis značajnih varijabli okruženja

Varijabla	Značenje
SHELL	ova varijabla označava ljusku koja se koristi pri prijavi na sustav
TERM	varijabla koja određuje vrstu terminala na kojem korisnik radi
PATH	ljuska traži naredbu po direktorijima navedenim u varijabli
HOME	određuje matični direktorij, i određene varijable se vežu na ovu varijablu
USER	programi mogu brzo pogledati korisnikov user ID i usporediti ga sa imenom korisničkog računa. Takav način usporedbe štedi vrijeme

`USER` - programi mogu brzo pogledati korisnikov user ID i usporediti ga sa imenom korisničkog računa. Takav način usporedbe štedi vrijeme.

`HOME` - određuje matični direktorij, i određene varijable se vežu na ovu varijablu.

Primjerice, naredba `cd` bez argumenata vraća mjesto rada u matični direktorij. Ako se promijeni varijabla home, to će imati za posljedicu da će i naredba `cd` utipkana bez argumenata vratiti mjesto rada u drugi direktorij.

```
$ echo $HOME
/home/studenti/direktorij
$ cd
$ pwd
/home/studenti/direktorij
$ unset HOME
$ export HOME=/home/studenti/direktorij/novidirektorij
$ echo $HOME
/home/studenti/direktorij/novidirektorij
$ cd
$ pwd
/home/studenti/direktorij/novidirektorij
```

PATH - govori u kojem direktoriju da traži, i kojim redosljedom. Korisno je dodati i više direktorija u kojima se traže naredbe. Zbog sigurnosnih razloga (virusi koji se nalaze u trenutnom direktoriju, a imaju ime naredbe) valja izbjegavati dodavanje . (trenutnog direktorija) u Path. Ako je već nužno ga dodati, dodaje se na kraj. Direktoriji se odvajaju dvotočkama, npr.

```
$ echo $PATH
/home/kreator/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/games:
/usr/local/sbin:/usr/local/bin:/usr/X11R6/bin:/root/bin
```

U primjeru se također mogu vidjeti razlike između PATH varijabli administratora i običnog korisnika. Administratori imaju uključene sbin direktorije (system binaries), te root direktorij. Dakako, postoji i standardna postavka PATH varijable, u slučaju da ona nije neposredno definirana od korisnika. Ta postvka ovisi od stroja do stroja, može izgledati ovako:

```
PATH=/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games
```

Za dodavanje dodatnog direktorija u varijablu PATH, bez ponovnog upisivanja citave varijable, moze se koristiti sljedeca sintaksa:

```
PATH=$PATH :/novi/direktorij
```

$PATH je stara varijabla PATH, kojoj se samo nadodaje direktorij /novi/direktorij na kraj.

10.6. Zadaci

1) Saznajte koju ljusku koristite.

2) Saznajte koje su ljuske na raspolaganju.

3) U slučaju da trenutno korištena ljuska nije bash promijenite ljusku u bash.

4) Saznajte koje su varijable u okruženju.

5) Definirajte novu varijblul A, i postavite ju na vrijednost 20. Zatim napravite novu

instancu ljuskse, i probajte saznati vrijednost varijable A. Kako treba definirati varijablu da bi bila vidljiva svim instancama ljuske?

6) Dodajte novi direktorij u PATH varijablu, zadrzavsi trenutni sadrzaj varijable.

7) Saznajte koje ste naredbe sve koristili tijekom rješavanja zadataka.

11. Korisne naredbe

11.1. Startup datoteke

Startup datoteke su one koje se izvršavaju pri logiranju korisnika na računalo. To su bash (Bourne-Again Shell), `ah`, `ksh`, `ash`, `zsh` konfiguracijske datoteke za ljuske. Slične datoteke je imao i MS-DOS (`autoexec.bat` i `config.sys`). Svaki put pri logiranju korisnika izvršavaju se bash datoteke. Naravno da korisnik može sam mijenjati, programirati već postojeće i nove datoteke.

Primjer za .bashrc datoteke:

```
$ .bashrc
User specific aliases and functions
Source global definitions
[-f /etc/bashrc ]; then
. /etc/bashrc
fi
```

11.2. Datum, vrijeme i kalendar

11.2.1. Naredba `time`

Struktura naredbe:

```
time [parametri] naredba [argumenti]
```

Naredba time ispisuje koliko je vremena potrebno da se izvrši neka druga naredba ili datoteka, što ovu naredbu čini jako praktičnom kod testiranja trajanja programa napisanih u bilo kojem programskom jeziku. Naredba se ispisuje u sljedećem obliku:

Real time: Vrijeme proteklo od pokretanja do prekida programa

User CPU time: Rezultat je suma *tms_utime* i *tms_cutime* polja

System CPU time: Rezultat je suma *tms_stime* i *tms_cstime* polja

U polje parametara koji su prikazani u se mogu upisati vrijednosti koje su prikazane u tablici 11.1 pomoću kojih se definira način ispisivanja rezultata naredbe `time`:

Tablica 11.1: Parametri narebe time

Parametri	Značenje
%E	realno vrijeme trajanja naredbe (u [sati :]minute:sekunde)
%e	vrijeme trajanja naredbe(u sekundama).
%S	vrijeme izraženo u CPU-sekundama koje mjeri trajanje procesa u kernel-modu
%U	vrijeme izraženo u CPU-sekundama koje mjeri trajanje procesa u user-modu
%P	Postotak korištenog CPU-a, koji se računa kao (%U + %S) / %

Ukoliko se uz naredbu time koristi sufiks "*-p"*, korisniku se daje mogućnost ispisa

rezultata u formatu *real %e user %U sys %S.*

Primjer:

```
$ time /bin/ls -al > /dev/null
/bin/ls -al > /dev/null  0.00s user 0.01s system 6% cpu 0.156 total
```

Zaključak:
Prikaz datoteka u direktoriju na ext2/3 datotečnom sustavu zahtijeva to veće vrijeme što ima više datoteka, ali porast nije linearan već polinoman zbog toga što su datoteke u dvostruko povezanoj listi.

11.2.2. Naredba `date`

Struktura naredbe:

```
date [parametri] [+format]
```

Za razliku od naredba `time`, `date` ispisuje i vrijeme i datum. Privilegirani korisnici (administratori) mogu mijenjati,postavljati datum i vrijeme.

Primjer:

```
 date
Thu Apr 29 13:41:35 EDT 2005
```

Neke opcije naredbe date prikazane su u tablici 11.2

Tablica 11.2. Opcije naredbe date

Opcije	Značenje
+format	ispisuje vrijeme i datum i proizvoljnom formatu
-r file	ispisuje vrijeme kada je zadana datoteka promijenjena
-s date	postavlja datum na željenu vrijednost
--help	ispisuje pomoć, sa svim opcijama korištenja naredbe

Primjer:

```
$ Date +"%A %j %n%k %p"
Thursday 128
13 PM
```

Naredba `date` može, koristeći sufikse u obliku +" parametri", specificirati način ispisivanja datuma na izlazu, odnosno format naredbe date. Neki od parametara prikayai su u tablici 11.3

Tablica 11.3. Neki parametri naredbe date

Parametri	Značenje
%A	puni naziv dana u tjednu
%B	puni naziv mjeseca
%c	lokalno vrijeme i mjesto
%D	datum u formatu (mm/dd/yy)
%m	mjesec (01..12)
%T	vrijeme u formatu (hh:mm:ss)
%x	lokalni datum (mm/dd/yy)
%Y	godina (1970...)

11.2.3. Naredba `cal`

Struktura naredbe:

```
cal [parametri] [[mjesec] godina]
```

Naredba ispisuje kalendar.

U polje parametara se mogu upisati sljedeće vrijednosti s sljedećim značenjima prikayanim u tablici 11.4:

Tablica 11.4. Parametri naredbe cal

Parametri	Značenje
-1	prikaz tekućeg mjeseca
-3	prikaza prijašnjeg/tekućeg/sljedećeg mjeseca
-s	prikaz nedjelje (Sunday) kao prvog dana u tjednu (default)
-m	prikaz ponedjeljka (Monday) kao prvog dana u tjednu
-j	prikaz Julijanskog datuma (dani su pobrojani od 1.1.)
-y	prikaz kalendara za tekuću godinu

Primjeri:

```
$ cal 11 1987
$ cal –m 11 1980
$ cal –y
```

11.3. Korištenje kalkulatora

11.3.1. Naredba bc

Pokretanjem naredbe ulazite u sučelje u kojem upisujete račun. Iz programa izlazite naredbom `^D`. Ako vam program otkaže poslušnost, spašavajte se tipkama ctr+d. Opcije naredbe *bc*prikazane su u tablici 11.5.

Tablica 11.5. Opcije naredbe bc

Parametri	Značenje
-i,	interaktivni mod
-l,	definiranje standardne math biblioteke
-s	procesira točno POSIX bc jezik
-v	ispisuje verziju bc-a
-h	ispisuje korištenje naredbe bc

Da bi koristili non-integer operacije nužno je pokrenuti program za ekstenzijom `-l`,`"bc -l"` da bi uključili mathlib tj. definirali željene naredbe.

Primjer:

```
$ bc -l
13-8
2+(sqrt(3)*e(2))
ln(e(1))
```

11.3.2. Naredba dc

Kalkulator koji radi sa postfix notacijom, te je puno kompliciraniji od prethodnog. Postfix notacija u ovom slučaju RPN notacija koristi se također na HP-ovim digitronima (HP48/HP49). Svaki argument mora biti u novom redu, i rezultat operacije se ne ispisuje, već je to potrebno zadati naredbom "p" (print). Posebno je koristan kod pretvaranja i računanja u različitim bazama. Program varijable(unos i rezultat) sprema na stog. Iz programa se izlazi naredbom `quit`. Opcije naraedbe dc prikazane su u tablici 11.6.

Tablica 11.6. Opcije naredbe dc

Opcije	Značenje
-f script-file	Mogućnost računanja skripte koja je uključena u datoteku
-e script	mogućnost definiranja skripte koju će procesirati kalkulator
-v	ispisuje verziju dc-a
-h	I spisuje korištenje naredbe dc

Način pisanja:prvi argument (enter), drugi argument (enter), operator (enter)

Primjeri:

```
$ dc
75
0.85
*
p
54
4
^
p
8
i
210
p
136
```

Sadržaj datoteke a možemo prikazati na sljedeći način.

```
$ cat a
75
0.25
/
p
$ dc -f a
300
```

11.4. Kreiranje pseudonima

11.4.1. Naredba `alias`

Naredbom alias stvaramo pseudonim, labelu za neku drugu često korištenu naredbu. Tako skraćujemo pisanje neke duge naredbe, ili želimo da se neka naredbe pokreće sa drugačijim parametrima.

Struktura naredbe:

```
alias [ime [naredba]]
```

Ako utipkamo samo naredbu alias, izlistat će se svi trenutno definirani pseudonimi. Jedan od korisnijih aliasa je `..` koji mijenja naredbu `cd ..`, dakle vraća vas u direktorij poviše.

Primjer:

Napraviti alias (pseudonim) za izlist svih (i skrivenih) datoteka.

```
$ alias lll="ls -al"
```

11.5. Stvaranje linkova

11.5.1. Naredba `ln`

Naredbom ln stvaramo pokazivač na željenu datoteku, dakle za razliku od alias gdje smo preimenovali naredbe, sada povezujemo file-ove, te im tako možemo pristupati pod drugačijim imenima. Pokazivači mogu biti meki (soft link) i tvrdi (hard link). Meki, simbolički pokazivač je samo pokazivač do mjesta gdje se nalazi datoteka, on pokazuje na ime, a ne na sadržaj datoteke. Koristi se da bi programe informirali gdje da potražuju datoteku, iako ona ne mora nužno postojati. Brisanjem originala mekani pokazivač više nije valjan. Tvrdi pokazivač veže datoteku. Implementirani su tako da je jedinstven sadržaj, ali se datoteke nalaze pod različitim imenima i na drugim lokacijama. Dakle više datoteka dijeli, pokazuje na isti sadržaj. Ako izbrišemo jedan pokazivač ništa se ne mijenja dok god postoji neki drugi koji pokazuje na sadržaj. Original ne postoji već su svi jednako vrijedni. Između ostalog meki pokazivač može pokazivati i na direktorije, pa i na one u drugom datotečnom sustavu, dok je sa tvrdim pokazivačima to nemoguće napraviti.

Struktura naredbe:

```
ln [parametri]  original [odredišno ime]
ln [parametri] original [odredišni direktorij]
```

Primjer:

```
$ ln –s fakultetelektrotehnikeiracunarstva_upis.pdf fer_u
```

11.6. Razmjena informacija među korisnicima

11.6.1. Naredba `write`

Naredbom write možemo komunicirati direktno sa drugim korisnicima. Relativno je jednostavna naredba, pokrećemo, naravno, upisom nje same i ime korisnika s kojim želimo komunicirati. Tada na njemu dođe poruka da ste zainteresirani za komunikaciju sa njim, a na njemu ili na njoj je da odluči hoće li prihvatiti. Drugi korisnik mora ponoviti isti postupak, tj. pokrenuti naredbu write zajedno sa vaše ime. Kad uspostavite vezu sa drugim računalom svaka linija koju utipkate šalje se drugoj osobi čim pritisnete enter.

Struktura naredbe:

```
write korisnik [tty]
```

`-tty` argument je terminal na kojeg želimo poslati text.

Primjer:

Prvo moramo omogućiti da drugi korisnici imaju dozvole pisanja na odredišnom terminalu na kojem se vi nalazite, inače nećete primati ni jednu poruku od drugog korisnika.

```
$ mesg y
is y
```

Najbolji način da saznate dali je korisnik s kojim želite komunicirati logiran na sistem je naredba who.

```
$ who | grep marica
marica         ttyAx   Apr 2 10:30
```

Ako vam se pojavi greška to je zbog toga što drugi korisnik nije omogućio primanje poruka naredbom "mesg y". S takvim korisnicima ne možete komunicirati. Naravno da isto vrijedi i za vas, u slučaju da imate isključeno primanje poruke, nitko ne može s vama komunicirati.

Sada pozivate korisnika da vam se pridruži

```
$ write marica
```

Na njegovom će ekranu stići poruka

```
Message from marica@netcom.com on ttyAx at 10:34…
```

Sada pričekajte da dugi korisnik upiše write ivica, tada će vam doći poruka:

```
Message from marica@netcom.com on ttyAx at 10:38…
```

Prekid razgovora radi se naredbom

```
$ ^d
```

11.7. Još korisnih naredbi

11.7.1. Naredba uname

Ovom naredbom ispisujemo najosnovnije informacije o računalu i operativnom sistemu. Pokretanjem naredbe bez dodatnih opcija ispisuje ime operativnog sustava (Linux)

Struktura naredbe:

```
uname [parametri]
```

Upisivanje sljedećih vrijednosti prikazanih u tablici 11.7 u polje parametara, ispisuju se sljedeće značajke:

Tablica 11.7: Parametri naredbe uname

Parametri	Značenje
-a	sve informacije o računalu
-s	ime jezgre (kernel)
-n	ispišu se ime korisnika u mrežnoj domeni
-m	ime računala (hardwersko)
-p	tip procesora
-i	hardware-ska platforma
-o	operacijski sustav

Primjer:

```
$ uname -a
Linux esa1 2.6.11.7-grsec #1 SMP Fri Apr 22 18:56:02 CEST 2005 i686
GNU/Linux
```

11.8. Naredba exec

Pokretanjem ove naredbe izvršavamo drugu naredbu (command)u trenutnoj ljusci, a ne kreira se novi proces. Naredbom `exec` "nestaje" trenutna ljuska.

Struktura naredbe:

```
exec [parametri]  [naredba]
```

Primjer:

```
$ exec ls
```

11.9. Naredba talk

Struktura naredbe:

```
talk osoba@poslužitelj
```

Ovom naredbom razgovaramo s drugim korisnikom. Osoba je korisničko ime neke osobe logirane na tvoj sustav. Ako je osoba logirana na nekom drugom sustavu pristupamo joj ovako: `osoba@poslužitelj`. Za razgovore sa osobom koja je logirana više puta, potrebno je naredbom “who” doznati na kojem je terminalu. Kad je veza uspostavljena korisnici mogu istovremeno pisati, a njihove poruke se vide u različitim prozorima.

Primjer:

```
$ talk ivica@host2
```

11.10. Naredba `uptime`

Ovom naredbom ispisujemo redom sljedeće:

- trenutno vrijeme
- koliko dugo sistem radi
- koliko je korisnika trenutno logirano (može uključivati istog korisnika više puta)
- prosjek opterećenja sistema (system load average)

Primjer:

```
$ uptime
10.45:12 up 39 min, 1 users, load average: 0.00, 0.00, 0.00
```

11.11. Zadaci

1) Izmjerite trajanje naredbi `ls HOME i ls /dev`, te izvedite zaključke o dobivenim rezultatima

2) Ispisati datum i vrijeme.

3) Saznajte kojeg ste dana (pon-ned) rođeni.

4) Izračunati `$ echo '(2+3)/4'|bc-l` koristeći ne interaktivan način:

5) Napraviti da se pri pokretanju bc kalkulatora uvijek uključi mathlib (`bc -l`).

6) Treba isprobati simboličke i tvrde linkove. Prvo treba u proizvoljnu datoteku u direktoriju $HOME povezati mekim pokazivačem sa datotekom */etc/passwd*, pregledati sadržaj, a zatim za istu datoteku napraviti tvrdi pokazivač.

7) Pokušajte komunicirati s nekim korisnikom (`write`).

8) Pokušajte razgovarati sami sa sobom (`talk`).

9) Pokrenite naredbe i pogledajte što sve ispisuje (`uname`, `uptime`).

12. Napredno korištenje Linux operacijskog sustava

12.1. Grafičko sučelje X

12.1.1. Osnove

Grafičko sučelje X je grafički sustav temeljen na klijent/poslužitelj paradigmi. Razvijen je na MIT-u 80-tih godina prošlog stoljeća za Unix operacijske sustave.

Svaki X sustav dijeli se na dva temeljna dijela – X poslužitelj i X klijent. Poslužitelj i klijent komuniciraju putem mreže upotrebom X protokola. Poslužitelj direktno kontrolira ulazne i izlazne uređaje, poput grafičkog zaslona, tipkovnice ili miša. On je zapravo bilo kakvo jače ili slabije računalo s direktnim pristupom sklopovlju. Klijenti, s druge strane, koriste ulazno/izlazne uređaje putem poslužitelja. Klijenti su ti koji vrše "stvarni" računalni posao, primjerice tekst procesor ili prikaz DivX filmova. Klijenti su aplikacije koje koriste X11 poslužitelj kako bi otvorili jedan ili više prozora te da bi prihvatili ulaz s tipkovnice ili miša. Valja spomenuti da X11 poslužitelj izvršava sva iscrtavanja na zaslonu, ne nužno samo "prozore". On je i prvi omogućio postojanje tzv. "thin" kijenata, kod kojih su klijenti samo prikaz, a poslužitelj radi razne proračune koji se samo prikazuju na klijentskim X11 poslužiteljima.

12.1.2. Uporaba X-a

X poslužitelj se u pravilu pokreće na računalu na kojega su spojeni monitor, tipkovnica i miš. Najčešća situacija je da se i klijent također pokreće na tom istom računalu. Kako su klijent i poslužitelj na istom računalu njihova međusobna komunikacija može ići preko mreže, no zbog brzine i specifične situacije, upotrebljavaju se druge (efikasnije) metode komunikacije. Međutim, rečeno je kako se X poslužitelju može pristupiti putem mreže pa je moguće pokrenuti klijenta na nekom udaljenom računalu. U većini slučajeva računalo na kojemu je poslužitelj i računalo na kojemu je klijent spojeni su putem nekakve brze veze, primjerice putem Etherneta ili ADSL-a. U principu moguće je koristiti i sporije komunikacijske veze, primjerice analogni modem. No, to se ne preporučuje jer je X protokol dizajniran za vrlo brze lokalne mreže i na sporim mrežama može biti neupotrebljiv za bilo kakav normalan rad.

Glavna prednost razdvajanja grafičkog sustava na poslužiteljski i klijentski koji međusobno komuniciraju putem dobro definiranog protokola je u tome što se programi (klijenti) mogu pokretati i izvršavati na puno jačim računalima.

Trenutno postoje tri najčešće implementacije X grafičkog sustava: Xorg i XFree86 te komercijalni AcceleratedX, s tim da je nekad postojao i Metro. Jasno, postoje i drugi X11 poslužitelji na non-linux platformama.

12.2. Editor `vi`

Vi je klasični uređivač teksta, a dolazi na komercijalnim Unix platformama. Postoje i njegove kasnije slobodne reimplementacije, uključujući `nvi`, `vim`, `vile` i `elvis`. Na Linuxu, naredba `vi` je uglavnom link na jedan od ovih programa.

Editor vi radi u 3 moda: naredbenom (1 – 2 slova naredbe, prefiksirane brojkom ili ne + kretanje), načinu unosa (isključivo unos teksta) i načinu za duge naredbe (počinju sa **:** i prikazuju se na ekranu). Takav načina rada čini vi atraktivnim editorom za korisnike koji razlikuju tekstualni unos od editiranja. Za korisnike koji editiraju dok tipkaju, emacs kao editor koji ne traži promjene načina rada pri unosu teksta bi bio bolji izbor.

Vi se zasniva na starijim linijskim editorima zvanim ex i ed. Korisnik može upotrijebiti snažne editorske sposobnosti unutar vi-a tipkajući dvotočku (:), unoseći naredbu i pritiskajući enter. Nadalje, možete smjestiti npr. ex naredbe u direktorij zvan ~/.exrc, koji vi čita na početku samog editiranja.

Jedna od najčešćih verzija vi-a na Linux sustavima je Bram Moolenaarov Vi Improved, tj. ***vim***. Na nekim Linux distribucijama, vim je default verzija vi-a i pokreće se kad pokrenete vi. Vim je promijenio neke osnovne mogućnosti vi-a, npr. omogućujući višestruke undo poteze.

Valjalo bi spomenuti i nekoliko osnovnih editora koje se može naći na standardnoj Linux distribuciji kao što su `pico`/`nano`, `elvis`, `emacs` i `joe`. Za početnike bi bio idealan `pico` zbog jednostavnosti korištenja, dok emacs s druge strane obiluje mogućnostima (npr. provjera pravopisa, čitanje mail-a, itd.). U usporedbi s vim-om, `emacs` je brz pri obradi malih količina teksta, ali sporiji pri obradi velikih datoteka, a i vim-ove naredbe su kompaktnije i u nekim slučajevima moćnije.

12.3. Elektronička pošta

12.3.1. Čitanje i pisanje poruka elektroničke pošte

U ovom poglavlju obrađujemo klijentske mail programe (MUA). Mail podrška nam omogućuje stvaranje, slanje, primanje i prosljeđivanje elektronske pošte. Program Mail ima 2 način rada: kreirajući način, u kojem kreirate poruku, i naredbeni način, u kojem upravljate svojim mailom.

Iako je mail moćan alat, može biti prilično "tricky" za neiskusnog korisnika. Najčešće je danas viđen u skriptama. Većina Linux distribucija uključuje nekoliko alata koji su bogatiji u mogućnostima i mnogo jednostavniji za korištenje: maileri ugrađeni u preglednike kao što je Mozilla, grafički mail programi distribuirani s GNOME-om (Evolution) i KDE-om (Kmail), i konzolni mail programi, od kojih je mutt, uz Gnus, danas vjerojatno najbolji. GNU Emacs editor također može primati i slati poštu. Spomenut ćemo i Thunderbird, kao prilično kvalitetni MUA.

12.3.2. Naredba `mail`

Za ulazak u interaktivni mod za čitanje mail-a, upišite:

```
$ mail
```

Za početak pisanja poruke korisniku, upišite:

```
$ mail korisnik
```

Upišite tekst poruke, jednu po jednu liniju, pritiskajući Enter na kraju svake. Za završetak poruke, upišite točku (.) na početak posljednjeg reda i pritisnite Enter.

```
$ mail -a 'From: Djed Mraz <djed.mraz@sjeverni.pol>'
-s 'Ovogodisnji pokloni' rudolf@sjeverni.pol < lista-poklona
```

Ovo će poslati mail sa Djed Mraz – ove e-mail adrese, sa Subjectom "Ovogodisnji pokloni", a sadržaj će neinteraktivno pokupiti iz datoteke "lista-poklona".

12.4. Rad s udaljenim računalima

12.4.1. Naredba `ssh`

SSH je kratica za „secure-shell“, protokol koji omogućava sigurno spajanje, prijenos podataka i tuneliranje X11 veze. Valja naglasiti da je SSH naziv i za protokol i za naredbu pa te dvije stvari treba razlikovati. SSH (protokol) je razvijen u Finskoj prije nekoliko godina. SSH naredbe su prvenstveno zamjena naredbama kao što su `rsh`, `rlogin` ili `rcp`, a omogućavaju snažnu enkripciju i provjeru integriteta kako u inicijalnom spajanju/rukovanju, tako i tijekom cijele komunikacije. SSH ne kriptira lozinku, već se ona u pravilu prenosi direktno do poslužitelja i odgovarajućeg mehanizma za baratanje lozinkama.

SSH se sastoji od dva dijela. SSH klijent se koristi na radnim stanicama S KOJIH se spajate, a SSH poslužitelj, ili daemon, se koristi na računalima NA KOJA se spajate. U mnogim slučajevima, i poslužitelj i klijent su instalirani na objema računalima, dopuštajući da se veze ostvaruju u oba smjera.

Kada instalirate softver i ispravno ga konfigurirate, bit ćete u mogućnosti pokrenuti ssh na klijent računalu. Upišite slijedeću naredbu, zamijenivši “korisnik” korisničkim imenom na udaljenom računalu, te “domaćin” imenom ili IP adresom udaljenog računala:

```
$ ssh -l korisnik domaćin
```

Primjetite da ako koristite isto ime na lokalnom i udaljenom računalu možete izostaviti dio `-l` korisnik.

Prvi put kad se spojite, bit ćete obavješteni da je “potpis“ udaljenog računala stavljen na listu poznatih domaćina. Ovaj “potpis“ je dodatni oblik sigurnosti – ako ikad u budućnosti neko drugo računalo zauzme mjesto prvotnog na koje se spajate i bude koristilo njegovo ime ili IP adresu, SSH program će vas o tome obavijestiti.

Nakon što unesete odgovarajuću šifru, dobit ćete obavijest ljuske udaljenog računala.

12.4.2. Naredba `wget`

Ta naredba služi za download sadržaja širom interneta, koristeći HTTP ili FTP protokole. Wget radi i sa sporim i nestabilnim vezama tako da nastavlja dohvaćati sadržaj sve do potpunog download-a. Nastavak prihvata sadržaja od zadnje pozicije radi na serverima koji to podržavaju. I HTTP i FTP obnavljanja download-a mogu biti vremenski obilježena, pa `wget` može vidjeti da li se udaljeni sadržaj u međuvremenu mijenjao i automatski obnoviti novu verziju, ako postoji.

Wget podržava proxy servere: ovo može ubrzati nastavak download-a sadržaja i omogućiti pristup iza vatrozida.

Pretpostavimo li da imate zeljeni URL, jednostavno upišite:

```
$ wget http://fly.cc.fer.hr/
-13:30:45--  http://fly.cc.fer.hr:80/
          => `index.html'
Connecting to fly.cc.fer.hr:80..     . connected!
```

```
HTTP request sent, fetching headers... done.
Length: 1,749 [text/html]
0K -> .
13:30:46 (68.32K/s) - `index.html' saved [1749/1749]
```

U slučaju nedovoljno stabilne ili brze veze, *wget* će pokušati dohvatiti sadržaj do kraja u nekoliko navrata. 20 je default-ni broj pokušaja završetka download-a. Jednostavno je promijeniti broj pokušaja na npr. 45:

```
$ wget --tries=45 http://fly.cc.fer.hr/jpg/flyweb.jpg
```

Sada možemo ostaviti *wget* da radi svoj posao u pozadini, uz zapisivanje svog napretka u log datoteku. Umjesto duljeg –tries pišemo –t:

```
$ wget -t 45 -o log http://fly.cc.fer.hr/jpg/flyweb.jpg &
```

Operator `&` na kraju omogućava da `wget` radi u pozadini. Za neograničen broj pokušaja, koristimo `-t inf`.

Korištenje FTP je također jednostavno. `wget` će se pobrinuti za korisničko ime i šifru (ovo vrijedi za tzv. anonimni ftp).

```
$ wget ftp://gnjilux.cc.fer.hr/welcome.msg
--23:35:55--  ftp://gnjilux.cc.fer.hr:21/welcome.msg
           => `welcome.msg'
Connecting to gnjilux.cc.fer.hr:21... connected!
Logging in as anonymous ... Logged in!
==> TYPE I ... done.  ==> CWD not needed.
==> PORT ... done.    ==> RETR welcome.msg ... done.
Length: 1,340 (unauthoritative)
    0K -> .
23:35:56 (37.39K/s) - `welcome.msg' saved [1340]
```

12.5. Arhiviranje podataka

12.5.1. Naredba `tar`

Kopira datoteke u ili obnavlja iste iz arhivskog medija. Drugim riječima, skuplja niz datoteka i sprema u jednu datoteku, tzv. "tok" bez sažimanja. Ako nije navedena opcija `-f` po defaultu arhivira na standardni izlaz, odnosno dearhivira sa standardnog ulaza. Ako je neka datoteka direktorij, tar djeluje na cijelo podstablo.

```
$ tar -cvf archive_name file1 file2 ...
```

Parametri naredbe tar prikazani su u tablici 12.1:

Tablica 12.1: Parametri naredbe tar

Znak	Značenje
c	potrebno za kreiranje nove datoteke
v	mogućnost prikaza imena svake datoteke koja se arhivira
f	potrebno za uporabu idućeg argumenta kao imena arhive
file1, file2, ...	imena datoteka koje se arhiviraju

Recimo da želite arhivirati datoteke: „freq.sas“, „plot.sas“ i „sort.sas“, i želite kreirati arhivsku datoteku zvanu „sasfiles.tar“. U ovom slučaju, *tar* naredba će biti:

```
$ tar -cvf sasfiles.tar freq.sas plot.sas sort.sas
```

12.5.2. Naredba `gzip`

Naredba gzip komprimira specificirane datoteke (ili pročitane sa standardnog ulaza) sa Lempel-Ziv kodiranjem (LZ77). Preimenuje kompresirane datoteke u filename.gz; Ako nema naveden argument, kompresira podatak sa standardnog ulaza, a sprema na standardni izlaz ako je navedena opcija –c.

Za otpakiravanje kompresirane datoteke koristimo naredbu `gunzip` ili `gzip -d`. Korisna je i naredba `split` za razlamanje datoteke ili arhive na više dijelova.

```
$ ls -al a.out
-rwxr-xr-x  1 kreator users 12884 May 11 23:30 a.out
```

Nakon naredbe *gzip*:

```
$ gzip a.out
$ ls -al a.out.gz
-rwxr-xr-x  1 kreator users 5244 May 11 23:30 a.out.gz
```

12.5.3. Naredba `bzip2`

Naredba za kompresiju i dekompresiju slična `gzip` naredbi, ali koristi drukčiji algoritam i metodu za kodiranje da bi dobila bolju kompresiju. Bzip2 (`bzip2`) naredba zamjenjuje sve datoteke sa svojom kompresiranom varijantom i .bz2 ekstenzijom u dodatku. Bunzip dekompresira svaki file kompresiran bzip2-om (ignorirajući druge datoteke). Bzcat (`bzcat`) dekompresira sve specificirane datoteke na standardni izlaz, a `bzip2recover` se koristi za pokušaj obnove podataka iz oštećenih datoteka.

```
$ ls -al a.out
-rwxr-xr-x  1 kreator users 12884 May 11 23:30 a.out
```

Nakon naredbe `bzip2`*:*

```
$ bzip2 a.out
$ ls -al a.out.bz2
-rwxr-xr-x  1 kreator users 5244 May 11 23:30 a.out.bz2
```

12.6. Filtriranje podataka, naredba awk

Naredba awk dozvoljava korisniku manipuliranje datotekama koje su strukturirane kao stupci podataka i stringova.

Može se koristiti za procesiranje podataka i automatizaciju aplikacije sastavljene od Unix naredbi. Sadrži i još mnoge funkcionalnosti koje su u širokoj upotrebi.

Postoje dva načina za pokretanje awk naredbe. Jednostavna awk naredba se može pokrenuti iz komandne linije. Kompleksnije operacije bi trebale biti napisane kao awk programi ("skripte") prema datoteci. Primjer:

```
$ awk 'uzorak {akcija}' ulazna_datoteka > izlazna_datoteka
```

Značenje: uzmi svaku liniju ulazne datoteke; ako linija sadrži uzorak primjeni akciju na liniju i rezultatnu liniju upiši u izlaznu datoteku.

Ako je uzorak izostavljen, akcija se primjenjuje na svaku liniju:

```
$ awk '{akcija}' ulazna_datoteka > izlazna_datoteka
```

Po default-u, awk u datotekama djeluje na stupce brojeva ili stringova koji su odvojeni praznim mjestima (razmacima ili tabovima), ali -F opcija se može koristiti ako su stupci odvojeni nekim drugim znakom

12.7. Prevođenje C programa, naredba gcc

GNU Compiler Collection, prevodi mnoge jezike (C, C++, Objective-C, Ada, FORTRAN, Java) u strojni kod.

GCC je skup programa koji prevode više programskih source datoteka: npr. C source datoteku (.c), assembler source datoteku (.s) ili preprocesorsku C source datoteku (.i). GCC sadrži i g++ (C++ prevodilac) kojeg korisnici mogu upotrebljavati za proizvoljno prevođenje željenog C/C++ programa. Ako sufiks u datoteci nije prepoznatljiv, pretpostavlja se da je datoteka objektna datoteka ili biblioteka. Gcc (gcc) uobičajeno poziva C preprocesor, prevodi procesni kod u asemblerski kod, asemblira ga, te ga, koristeći link editor, povezuje sa različitim sistemskim i inim bibliotekama.. Ovaj se proces može zaustaviti korištenjem -c, -S ili -E opcija. Koraci također mogu odstupati ovisno o jeziku koji se kompajlira. Po default-u, izlaz je smješten u a.out. U nekim slučajevima, gcc generira objektnu datoteku uz -o sufiks i odgovarajuće osnovno ime.

```
gcc <ime_ulazne_datoteke> -o <ime_izlazne_datoteke>
```

Preprocesorske i povezivačke opcije navedene u gcc naredbi prosljeđuju se tim alatima u samom izvođenju. Napomena: *gcc* je GNU oblik cc-a, na većini Linux sustava, naredba cc će pozvati gcc.

Kompajliranje .c i .cpp datoteka je jednostavno i istovjetno (osim ključnih riječi – gcc za C datoteke i g++ za C++ datoteke). Primjer kompajliranja C++ datoteke:

```
$ g++ -Wall hello.cc -o hello
```

Izvršna datoteka se jednako pokreće za C i C++ datoteke:

```
$ ./hello
Hello, world!
```

13. Literatura

1) S. Figgins, E. Siever, A. Webber: *Linux in a Nutshell, 4th Edition*, O'Reilly, 2003.

2) D. Taylor, J. C. Armstrong: *Teach Yourself UNIX in 24 Hours*, Sams Publishing, Indianapolis, 1997.

3) J. Tacket, S. Burnet: *Using Linux*, Que Corporation, Indianapolis, 1998.

4) M. Žagar: *UNIX i kako ga koristiti*, "Antonić", Zagreb, 1995.

5) Bradford learning, *LPI 101 Preparatory Course*

Osnove Linuxa
Autor: Igor Smud
Korisnici u Linux operacijskom sustavu
Autor: Luka Lovošević
Rad s datotekama i direktorijima
Autor: Marko Salkić
Struktura datotečnog sustava u Linuxu
Autor: Jelena Vučak
Kreiranje datoteka
Autor: Maja Varga
Vlasništvo i dozvole
Autor: Marko Čavka
Pretraživanja, filtri i cjevovodi
Autor: Goran Delač
Zamjenski znakovi i regularni izrazi
Autor: Maja Penović
Rad s procesima
Autor: Marko Tošić
Ljuske
Autor: Vedran Marinkov Krecl
Korisne naredbe
Autor: Mario Kozina
Napredno korištenje Linux operacijskog sustava
Autor: Slaven Žilić