



**Sveučilište J. J. Strossmayera u Osijeku**

**Fakultet elektrotehnike, računarstva i  
informacijskih tehnologija**

Kneza Trpimira 2b

HR-31000 Osijek

[www.ferit.unios.hr](http://www.ferit.unios.hr)

---

## **Laboratorijska vježba I:**

Pregled UNIX/Linux operacijskih sustava i osnovna upotreba  
Bash ljuske

## Sadržaj

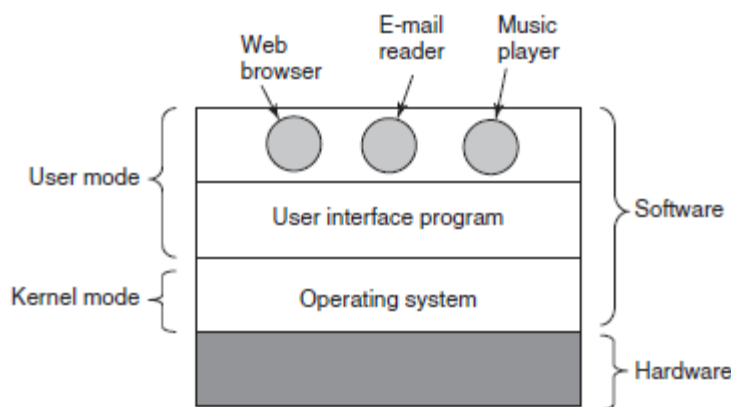
1. UVOD .....	2
1.1. Ciljevi i funkcije operacijskog sustava .....	2
1.2. Zašto učiti operacijske sustave? .....	5
1.3. Što se dogodi kada se pokrene računalo? .....	6
1.4. Povijest operacijskih sustava .....	6
1.5. Podjela operacijskih sustava .....	7
2. UNIX/LINUX OPERACIJSKI SUSTAVI .....	9
2.1. Osnovni koncepti UNIX/Linux operacijskih sustava .....	10
2.2. Arhitektura UNIX/Linux operacijskih sustava .....	10
2.3. Mrežni pristup udaljenom Linux poslužitelju .....	13
3. UVOD U LJUSKU OPERACIJSKOG SUSTAVA LINUX .....	14
3.1. Osnovne naredbe ljuske .....	14
3.2. Jednostavno kretanje kroz datotečni sustav .....	15
3.3. Jednostavan rad s datotekama .....	16
3.4. Naredbe za olakšano korištenje znakovnog sučelja .....	18
3.5. Naredbe za pretraživanje Linux priručnika .....	18
4. ZADACI ZA SAMOSTALNI RAD .....	20
5. LITERATURA .....	22

# I. UVOD

Suvremeno računalo se sastoji od jednog ili više procesora, glavne memorije, diskova, pisača, tipkovnice, miša, zaslona, mrežnih uređaja i raznih drugih ulazno-izlaznih uređaja. Upravljanje svim tim komponentama i njihova optimalna uporaba je iznimno zahtjevan posao. Iz tog razloga su računala opremljena slojem softvera koji se zove operacijski sustav, a čiji je zadatak pružiti korisničkim programima bolji i jednostavniji model računala te upravljanje spomenutim resursima.

Svi današnji korisnici računala imaju neko iskustvo u radu s operacijskim sustavima, kao što su Windows, macOS ili Linux. Program s kojim korisnici komuniciraju obično se naziva ljska, a postoji u obliku tekstualnog ili grafičkog sučelja (GUI). Međutim, niži slojevi operacijskog sustava, koji odrađuju gotovo cijeli posao, se uglavnom promatraju kao „crna kutija“. Većina računala ima dva načina rada: način jezgre (engl. **kernel/supervisor mode**) i korisnički način (engl. **user mode**). Operacijski sustav, kao najosnovniji softver, pokreće se u načinu jezgre. U ovom načinu rada ima cjelokupni pristup sklopovlju i može izvršavati strojne instrukcije. Ostali softveri rade u korisničkom načinu, u kojem je dostupan samo podskup strojnih instrukcija. Bitna razlika između operacijskog sustava i aplikacijskog softvera je što se aplikacijski softver lako može zamijeniti (npr. Internet preglednik, čitač e-pošte,...), dok je to teško napraviti s dijelovima operacijskog sustava.

U ovoj laboratorijskoj vježbi ćemo se ukratko dotaknuti nekoliko ključnih aspekata operacijskih sustava, pojasniti njihovu ulogu, povijest, podjelu te arhitekturu. Detaljnije ćemo se upoznati s građom operacijskih sustava UNIX i Linux koje ćemo koristiti za praktičan prikaz osnovnih koncepata operacijskih sustava u idućim laboratorijskim vježbama. Kako bi se dotaknuli jezgre operacijskog sustava Linux i njenih funkcionalnosti, moramo pak imati osnovno znanje o korištenju tog operacijskog sustava na razini korisnika. Ovom laboratorijskom vježbom započinjemo s korištenjem ljske operacijskog sustava Linux, gdje ćemo prikazati osnovne naredbe za snalaženje u tekstualnom sučelju. Kroz prve tri laboratorijske vježbe ćemo sažeto prikazati način korištenja najpoznatije ljske – **Bash ljske**.



**Slika 1.** Slojeviti prikaz hardvera i softvera računala

## I.1. Ciljevi i funkcije operacijskog sustava

Operacijski sustav je program koji kontrolira izvršavanje aplikacijskih programa i služi kao sučelje između njih i računalnog sklopovlja. Ciljevi operacijskog sustava su trojaki:

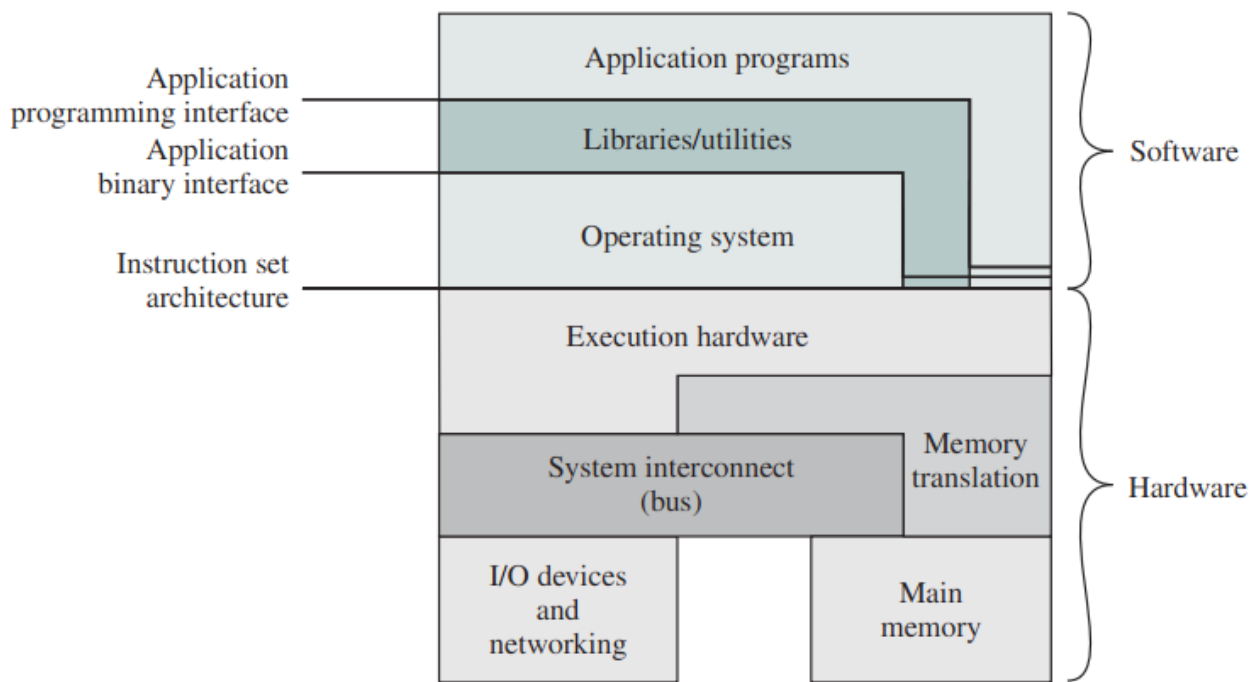
- Pogodnost – operacijski sustav čini računala pogodnijim za korištenje.

- Efikasnost – operacijski sustav omogućuje efikasnije korištenje resursa računalnog sustava.
- Sposobnost rasta – operacijski sustav bi trebao biti konstruiran tako da dozvoljava efektivan razvoj, testiranje i uvođenje novih funkcionalnosti sustava bez ometanja korisničkih programa.

Funkcije operacijskog sustava se promatraju iz dva aspekta, koji će se pojasniti malo detaljnije u sljedećim potpoglavljima.

### 1.1.1. Operacijski sustav kao sučelje između korisnika i računala

Računalno sklopovlje i programska podrška se mogu prikazati slojevito u hijerarhijskom obliku, kao što je prikazano na slici 2. Krajnjeg korisnika računala uglavnom ne zanimaju detalji funkcioniranja sklopovlja, već određeni aplikacijski programi. Aplikacija se može izraziti programskim jezikom, a razvija ju aplikacijski programer. Ukoliko bi programer morao razvijati aplikaciju kao skup strojnih instrukcija koje su u potpunosti zadužene za upravljanje sklopovljem, suočio bi se s ogromnim izazovom. Kako bi se olakšao ovaj zadatak, razvijen je skup sistemskih programa, često zvanim i pomoćnim programima (engl. *utilities*) ili bibliotekama (engl. *libraries*). U njima su sadržane često korištene funkcije koje pomažu u kreiranju programa, upravljanju datotekama i kontroliranju I/O uređaja. No, najvažnija kolekcija sistemskih programa je pak operacijski sustav, jer on „maskira“ detalje računalnog sklopovlja od programera i pruža mu pogodno sučelje za njegovo korištenje.



**Slika 2.** Računalno sklopovlje i programska podrška

Poznata je činjenica da se svaki aplikacijski program mora svesti na „skup nula i jedinica“ kako bi u konačnici bio pokrenut na računalnom sklopovlju. Slika 2 također navodi tri ključna sučelja koja omogućavaju taj prijelaz:

- **Arhitektura skupa instrukcija** (engl. *instruction set architecture (ISA)*) – ISA definira repertoar strojnih instrukcija koje računalno može izvršiti. Ovo sučelje je granica između hardvera i softvera, a oba sloja mu mogu direktno pristupiti.

- **Binarno aplikacijsko sučelje** (engl. *application binary interface (ABI)*) – ABI definira standarde za binarnu portabilnost programa. Aplikacijski programi koji se svedu na binarni oblik se lakše mogu pokretati na različitim računalnim arhitekturama. ABI je standard utemeljen od strane operacijskog sustava i sklopovlja, a kojeg se trebaju pridržavati prevoditelji programskog koda. Pojednostavljeno, ABI je kompajlirana verzija API-ja, koja pomaže prevoditeljima programskog koda u generiranju strojnog koda prikladnog za izvođenje na određenoj ISA arhitekturi. ABI čini softver neovisnim o hardveru.
- **Programsko aplikacijsko sučelje** (engl. *application programming interface (API)*) – API programu daje pristup računalnim resursima i uslugama dostupnima u ISA pomoću **sustavskih poziva**. Korištenjem API-ja aplikacijski program se može lako prenositi i ponovno prevoditi na različitim računalnim arhitekturama.

Ukratko, usluge operacijskog sustava koje on pruža kao sučelje između korisnika i računala olakšavaju sljedeće poslove:

- Razvoj programa – operacijski sustav nudi skup usluga, kao što su editori i debuggeri koji pomažu prilikom programiranja. Te usluge uglavnom nisu temeljni dio operacijskog sustava, ali dolaze uz njega i referira ih se kao alati za razvoj aplikacijskih programa.
- Izvršavanje programa – operacijski sustav rukuje raspoređivanjem zadataka i memorije prilikom izvršavanja instrukcija nekog programa.
- Pristup I/O uređajima – svaki I/O uređaj zapravo zahtijeva specifičan set instrukcija i kontrolnih signala tijekom svog rada. Međutim, operacijski sustav pruža uniformno sučelje koje skriva te detalje od programera i omogućava jednostavnije i općenite mehanizme za čitanje i pisanje.
- Pristup sustavu – operacijski sustav kontrolira pristup resursima sustava, tj. pruža slojeve zaštite od neautoriziranih korisnika te rješava konflikte prilikom istovremenog pristupa.
- Detekcija pogrešaka – operacijski sustav sadrži mehanizme za reagiranje na velik broj mogućih pogrešaka i sa strane sklopovlja (hardverski kvar, greška u memoriji,...) i sa strane programske podrške (dijeljenje s 0, pristup zabranjenoj memorijskoj lokaciji,...).
- Vođenje statistike – dobar operacijski sustav prikuplja statistike o korištenju različitih resursa te nadzire performansama.

### 1.1.2. Operacijski sustav kao upravitelj resursa

Računalo je skup resursa za prijenos, pohranu i procesiranje podataka i za kontrolu spomenutih funkcija. Često se može čuti rečenica kako operacijski sustav upravlja resursima računala. Pri tome se stječe dojam kako je operacijski sustav nekakav vanjski mehanizam, odvojen od računala, koji nadgleda i kontrolira njegov rad. Međutim, operacijski sustav, kao i ostali aplikacijski programi, čini dio softvera koji se mora izvršiti na procesoru. Osim toga, operacijski sustav često prepušta kontrolu te ovisi o procesoru kako bi ju ponovno stekao. Kao i svi ostali programi, operacijski sustav pruža skup instrukcija procesu na izvršavanje. Ključna razlika je namjera tih instrukcija, tj. operacijski sustav usmjerava procesor na korištenje drugih resursa i upozorava na vrijeme izvršavanja drugih programa. No, kako bi procesor izvršio te „ostale programe“, operacijski sustav mu mora prepustiti kontrolu i zadržati ju tek toliko da ga usmjeri na drugi zadatak.

Zbog svega navedenog, velik dio operacijskog sustava je smješten u glavnoj memoriji računala. Taj dio čine jezgra, koja sadrži najčešće funkcije operacijskog sustava i ostali dijelovi po potrebi. Ostatak glavne memorije čine korisnički programi i podaci. Operacijski sustav odlučuje kada program u izvršavanju smije koristiti I/O uređaj te kontrolira pristup datotekama. Procesor je također računalni resurs, pa operacijski sustav i za njega mora

odrediti koliko vremena smije posvetiti određenom programu. Na kraju možemo zaključiti kako je operacijski sustav tek program kojeg procesor izvršava, no koji mu pritom ukazuje na njegove sljedeće zadaće.

## 1.2. Zašto učiti operacijske sustave?

Smisao učenja operacijskih sustava nije u učenju kako kreirati novi operacijski sustav od nule, već naučiti kako operacijski sustav radi. Sa programerskog stajališta, da biste u potpunosti razumjeli kako vaš program funkcionira, trebali biste imati opće razumijevanje onoga što se događa iza kulisa. Razumijevanje kako funkcioniraju datotečni sustav, memorija ili procesor vas definitivno može učiniti boljim programerima. Na kraju krajeva, unutarnji rad računala bi trebao biti zanimljiv svakom studentu računarstva!

S druge strane, postoje poslovi koji **eksplicitno** zahtijevaju određenu razinu znanja o operacijskim sustavima. Najčešći takvi poslovi su:

1. **Sistemske programer** – programeri vlastitih operacijskih sustava ili njihovih nadogradnji. Uglavnom se fokusiraju na razvoj jezgre, upravljačkih programa, kompajlera i debuggera. Programiraju u assembleru, C i C++ programskim jezicima.
2. **Linux softverski inženjer** – održavaju i unaprjeđuju glavne komponente operacijskog sustava te razvijaju prilagođene verzije Linuxa. Moraju dobro poznavati Linux jezgru i infrastrukture vezane uz konfiguraciju operacijskog sustava (upravljački programi, servisi, procesi). Također moraju imati znanje o Linux C/C++ programiranju (kompajleri, biblioteke, alati) te dobro poznavati skriptne jezike ljuske (Bash).
3. **Embedded developer** – razvijaju softver za ugradbene računalne sustave, primarno u C/C++ programskom jeziku te rade na sigurnosno-kritičnim operacijskim sustavima. Moraju dobro poznavati principe operacijskih sustava te razvoj višejezgrenih i višenitnih aplikacija na Linux OS-u ili RTOS-u.
4. **Sistemske administrator** – zadaće su im raznolike: provjera logova poslužitelja; izrada zakrpa i konfiguriranje poslužitelja (lokalno ili u oblaku); razvoj, praćenje i testiranje sigurnosne kopije; pisanje build i deploy skripti; priprema računalnih komponenti za nove poslužitelje; instaliranje Linux distribucija.
5. **DevOps** – uspostavljaju, održavaju i unaprjeđuju procese razvoja softvera, održavaju poslužitelje, automatiziraju upravljanje okolinama, nadziru i prate aplikacije, automatiziraju testiranje. Moraju imati napredno iskustvo u radu s Linux OS-om, skriptnim jezikom Bash i s nekim programskim jezikom.
6. **Software developer** – osim samog pisanja programskog koda, pripremaju i okoline za razvoj aplikacije u nekom od programskih jezika (npr. PHP, Python, Java, Ruby,...) pri čemu je potrebna konfiguracija sustavskih alata operacijskog sustava na kojem se radi.
7. **Etički haker** - identificiraju slabosti u računalnim sustavima i mrežama putem kojih mogu steći pristup u računalo. Moraju imati napredno iskustvo u radu s Linux OS-om te poznavati mrežne tehnologije.

Laboratorijske vježbe iz ovog kolegija sastavljene su tako da usvojite sljedeće vještine iz domene operacijskih sustava:

- Rad u tekstualnom sučelju operacijskog sustava Linux
- Pisanje Bash skripti
- Koncepte paralelnog programiranja
- Razumijevanje upravljanja resursima
- Analiza performansi unutar OS-a

### I.3. Što se dogodi kada se pokrene računalno?

- 1) Na matičnoj ploči se nalazi ROM memorija u kojoj je smješten BIOS/UEFI **firmware**.
- 2) Prvi korak nakon paljenja računala je učitavanje i izvršavanje BIOS/UEFI **firmware**-a. Taj program započinje s inicijalizacijom sklopovlja, tj. prikuplja informacije o sklopovlju spojenom na matičnu ploču te prosljeđuje te informacije u RAM memoriju kako bi ih procesor mogao pročitati. BIOS/UEFI **firmware** je strojni kod pisan kombinacijom assemblera i programskog jezika C, a kompajliran je za specifičnu računalnu arhitekturu na kojoj se nalazi.
- 3) Instrukcijski registar (engl. *instruction register*) procesora čita sadržaj RAM memorije i započinje izvršavati **firmware**.
- 4) **Firmware** upućuje procesor na spajanje s sklopovljem te pokreće program zvan **bootloader** koji preuzima upravljanje i ponaša se kao medijator između sklopovlja i operacijskog sustava.
- 5) **Bootloader** se učitava u radnu memoriju te skenira sve medije na kojima se nalaze izvršne slike operacijskih sustava. Zatim ispisuje listu operacijskih sustava instaliranih na sklopovlju. Korisnik može izabrati jedan od njih.
- 6) **Bootloader** prema zadanom prioritetu odabire medij za pokretanje te u radnu memoriju učitava izvršnu datoteku jezgre operacijskog sustava. Ovaj proces se naziva dizanje operacijskog sustava (engl. *booting*). Izvršna slika operacijskog sustava se može nalaziti u prvom bloku medija za pokretanje ili na nekoj specifičnoj particiji.
- 7) Kako bi jezgra operacijskog sustava dobila saznanje o arhitekturi računala na kojoj se pokreće, povezuju se izvršna slika **bootloader**-a i jezgre te se stvara datoteka *kernel.bin*. Povezivanje odrađuje program *linker*, te na taj način operacijski sustav dobiva informaciju o adresama registara procesora.
- 8) Operacijski sustav se potpuno učitava te počinje upravljati cjelokupnim softverom.
- 9) Pokreću se dodatne usluge (servisi u pozadini) koji su dio ili proširenje operacijskog sustava. Najčešće su to *firewall*, antivirusna zaštita, ažuriranje sustava i sl.
- 10) Na kraju se pokreću u usluge koje su dio korisničkog proširenja sustava, kao što su: programi za pohranu podataka u oblaku (*Dropbox*, *OneDrive*,...), programi za prilagodbu ponašanja dijelova sustava (zaslon, zvuk, ...), komunikacijski alati (*Skype*) i ostali.

### I.4. Povijest operacijskih sustava

Operacijski sustavi su povijesno bili usko vezani za arhitekturu računala na kojima rade, tj. svaka generacija računala dolazi s karakterističnim operacijskim sustavom. Generacije računala:

1. Prva generacija (1945-55): Vakuumske cijevi
  - Z3, Colossus, Mark I, ENIAC
  - Mala grupa ljudi dizajnira, razvija, programira, upravlja i održava pojedinačne strojeve
  - Programiranje u strojnom jeziku ili zavarivanje električnih krugova
2. Druga generacija (1955-65): Tranzistori i serijski sustavi
  - Mainframe računala
  - Programiranje u assembleru ili FORTRANU te zapisivanje na bušene kartice
  - Serijski sustavi – lista programa pohranjenih na magnetskog vrpce
3. Treća generacija (1965-1980): Integrirani krugovi i multiprogramiranje
  - IBM 360 IC s općom arhitekturom i setom instrukcija za „sva“ računala
  - Operacijski sustav OS/360 napisan u assembleru; nedostatak je što je bio usko vezan uz integrirani krug i morao je raditi na svim računalima, od malih ka velikim

- Multiprogramiranje – raspodjela više I/O operacija istovremeno
  - *Spooling* – mogućnost automatskog učitavanja novog posla čim stari završi
  - Pojava potrebe za dijeljenjem vremena (engl. *timesharing*) – varijanta multiprogramiranja u kojoj se resursi dinamički raspodjeljuju između više različitih korisnika ovisno o njihovim potrebama i stanju poslova
  - **MULTICS** sustav razvijan od MIT, Bell Labs i General Electric – stroj koji simultano podržava stotine korisnika; projekt nije bio uspješan, ali je imao značajan utjecaj na operacijske sustave koji zatim slijede (UNIX i njegove izvedenice)
  - 1969. godine Ken Thompson iz Bell Labs razvija jednokorisničku verziju MULTICSA – **UNIX**. Izvorni kod je postao dostupan i ubrzo se razvijaju 2 velike verzije: **System V** (AT&T) i **BSD** (Berkeley)
  - IEEE razvija standard za pisanje programa koji se trebaju pokretati na UNIX-u – **POSIX**
  - 1987. godine dolazi do razvoja manjeg klona UNIX-a, nazvanog **MINIX**, poglavito za edukacijsku namjenu.
  - 1991. godine, Finac Linus Torvalds u želji za razvojem besplatne verzije MINIX-a, razvija operacijski sustav **Linux**
4. Četvrta generacija (1980-danas): Osobna računala
- Razvoj LSI integriranih krugova i pojava osobnih računala (npr. IBM PC)
  - Za svoje osobno računalo IBM PC, tvrtka IBM kontaktira Bill Gatesa za kupnju njegovog BASIC interpretera, a zatim od njega traže i razvoj prikladnog operacijskog sustava (nakon odbijenice od tvrtke Digital Research)
  - Gates kupuje **DOS** operacijski sustav od tvrtke Seattle Computer Products, povezuje ga sa svojim BASIC interpreterom i modificira u novi operacijski sustav **MS-DOS**
  - Krajem 1960-tih, Doug Engelbart sa Stanforda izumio grafičko sučelje (**GUI**) s prozorima, ikonama, menijem i mišem
  - 1983. g. vlasnik tvrtke Apple Steve Jobs inkorporira GUI i razvija **Lisa**, a 1998.g. i **Macintosh** osobna računala
  - 1999. godine, Apple prisvaja Mach mikrojezgru temeljenu na UNIX-u i razvija **MAC OS X** operacijski sustav
  - Od 1985. do 1995. g. razvija se operacijski sustav **Windows** kao grafičko sučelje na vrhu MS-DOS operacijskog sustava
  - Nakon 1995. g. **Windows 95** izlazi kao samostalni operacijski sustav. Nakon toga se još pojavljuju verzije Windows 98, Windows NT, Windows Me, Windows 2000, Windows XP, Windows Vista, Windows 7, 8, 10, ...
5. Peta generacija (1990-danas): Mobilna računala
- U prvom desetljeću njihova nastanka, većina mobilnih uređaja je pokretalo **Symbian** operacijski sustav (Samsung, Sony Ericsson, Motorola i Nokia)
  - 2002. izlazi **Blackberry OS**, a 2007. **iOS** od tvrtke Apple
  - 2011. Nokia napušta Symbian i prelazi na **Windows Phone**
  - 2008. Google objavljuje **Android** temeljen na Linuxu, koji ubrzo postaje najkorišteniji operacijski sustav za mobilne uređaje

## 1.5. Podjela operacijskih sustava

Dvije osnovne grupe operacijskih sustava prema broju korisnika:

- jednokorisnički (desktop) – DOS, Windows 3.x/95/98/Me/XP, Linux, MAC OS,...



- višekorisnički (network) – UNIX, Windows (Server verzije sve, od kućnih NT i noviji, iako se Vista smatra prvim pravim – UAC), Nowell Netware, Mainframe NOSs (Digital Equipment VMS, Hewlett-Packard MPE, IBM MVS,...)

Podjela prema ulogama računala, tzv. zoološki park operacijskih sustava (engl. *The operating systems zoo*):

- poslužitelji: datotečni, aplikacijski, database, mail, web...
- desktop računala: multimedija, CAD radne stanice, uredska računala, računala za igru...
- ugrađeni (engl. *Embedded*) uređaji: mp3 playeri, automobili, uređaji posebne namjene...
- real-time računala: upravljanje RT procesima – CNC strojevi za obradu materijala...

Sve verzije modernih mrežnih operacijskih sustava podržavaju

- višekorisnički rad
- višezadaćnost
- raspodijeljeno procesiranje
- visok stupanj sigurnosti

## 2. UNIX/LINUX OPERACIJSKI SUSTAVI

U današnje vrijeme uglavnom izdvajamo tri osnovna operacijska sustava za osobna računala, sa svojim distribucijama: Linux, Windows i macOS. Kroz laboratorijske vježbe ćemo se fokusirati na Linux te kratko dotaknuti operacijskog sustava Windows. Linux je član velike obitelji operacijskih sustava nalik UNIX-u (engl. *UNIX-like operating systems*). Inicijalno ga je razvio Linus Torvalds 1991 godine kao operacijski sustav za IBM-kompatibilna osobna računala temeljena na Intel 80386 mikroprocesorima. UNIX operacijski sustav je razvijen 1969. godine u AT&T Bell laboratoriju. Razvojni tim su činili Ken Thompson, Dennis Ritchie, Douglas McIlroy i Joe Ossana. U prvoj verziji, UNIX je napisan kompletno u assembleru, a 1973. Dennis Ritchie ga prepisuje u programskom jeziku C. Dostupnost tog jezika visoke razine čini portabilnost operacijskog sustava Linux mnogo lakšom. Kroz nekoliko godina, UNIX postaje široko upotrijebljen u akademskim institucijama i poslovnim tvrtkama. Međutim, Bell laboratorij uskoro donosi odluku o prodaji operacijskog sustava UNIX kao komercijalnog proizvoda. U međuvremenu, student sveučilišta u Helsinkiju, Linus Torvalds, frustriran licencama operacijskih sustava temeljenih na UNIX-u, započinje s razvojem vlastite jezgre koja s vremenom postaje poznatija pod nazivom Linux. Linus je razvoj vlastite jezgre temeljio na jezgri UNIX-a, no po razvoju konačnog proizvoda odlučuje ga besplatno objaviti i učiniti otvorenim (engl. *open source*). Ubrzo su se tom projektu pridružili i drugi programeri i sve se više širi zajednica koja razvija i poboljšava nove verzije i distribucije operacijskih sustava temeljenih na Linux jezgri. Dovoljno je reći da je u razvoju verzije 4.1. Linux jezgre sudjelovalo gotovo 14 tisuća programera. Izvorni kod Linux jezgre može se pronaći i preuzeti na: <http://www.kernel.org>.

**Tablica 1.** Najpoznatiji proizvođači računalnog sklopovlja

Proizvođač sklopovlja	Verzija Linux OS-a	Ostali specifični OS-ovi
<b>IBM</b>	AIX	MVS, VM
<b>Hewlett Packard (HP)</b>	HP-UX	MPE, NonStop OS
<b>Digital Equipment (Compaq)</b>	Tru64, Ultrix, HP-UX	VMS
<b>Sun Microsystems</b>	Solaris	SunOS
<b>Intel</b>	Solaris	NetWare, Windows OS

Velik broj neprofitnih i profitnih organizacija i tvrtki razvija vlastite verzije Linux operacijskog sustava s različitim kombinacijama korisničkih programa, alata i ostalog softvera. Kombinacija OS-a (karakteriziranog jezgrom) i programske podrške naziva se distribucija.

**Tablica 2.** Najpoznatije distribucije operacijskog sustava Linux

Distribucija	URL
<b>LinuxMint</b>	<a href="https://linuxmint.com/">https://linuxmint.com/</a>
<b>Ubuntu</b>	<a href="http://www.ubuntu.com/">http://www.ubuntu.com/</a>
<b>Debian (prije Debian GNU/Linux)</b>	<a href="http://www.debian.org/">http://www.debian.org/</a>
<b>fedora</b>	<a href="https://getfedora.org/">https://getfedora.org/</a>
<b>openSUSE</b>	<a href="https://www.opensuse.org/">https://www.opensuse.org/</a>
<b>Archlinux</b>	<a href="https://www.archlinux.org/">https://www.archlinux.org/</a>
<b>CentOS</b>	<a href="https://www.centos.org/">https://www.centos.org/</a>
<b>FreeBSD</b>	<a href="http://www.freebsd.org/">http://www.freebsd.org/</a>

Na današnjem tržištu postoji mnogo UNIX i Linux distribucija, no većina dijele fundamentalne ideje i značajke dizajna. U tom smislu, moguće je usporediti Linux s nekom distribucijom UNIX-a. Od 2.6. verzije Linux nastoji biti podudaran s standardom IEEE POSIX, što znači da se većina UNIX programa može kompajlirati i pokrenuti na Linux sustavima uz vrlo male izmjene izvornog koda. Nadalje, Linux uključuje sve značajke modernog UNIX-a, kao što su virtualna memorija, virtualni datotečni sustav, laki procesi, UNIX signali, međuprocena komunikacija i slično.

S procijenjenim brojem instalacija od nekoliko desetaka milijuna, ljudi koji su navikli na određene značajke u modernim operacijskim sustavima isto očekuju i od Linux-a. U tom pogledu, povećava se i potražnja za Linux programerima.

## 2.1. Osnovni koncepti UNIX/Linux operacijskih sustava

Osnovni koncepti UNIX/Linux operacijskih sustava su:

- Višekorisnički rad – mogućnost **istovremenog** i **neovisnog** izvršavanja nekoliko aplikacija koje pripadaju različitim korisnicima. Istovremeno znači da se aplikacije istovremeno natječu za različite vrste resursa, kao što su procesor, memorija ili disk. Neovisno znači da svaka aplikacija može izvršavati svoje zadatke bez brige o radu drugih aplikacija. Višekorisnički rad je omogućen pomoću nekoliko značajki, kao što su: autentikacijski mehanizmi za identifikaciju korisnika, zaštitni mehanizmi od malicioznih ili logički-pogrešnih programa te nadziranje i statističko praćenje rada računala
- Korisnici i grupe – svaki korisnik ima privatni prostor na stroju, a međusobno se razlikuju putem identifikatora. Više korisnika može pripadati grupi. Postoji specijalni korisnik koji ima gotovo sve ovlasti, a naziva se *root* ili *superuser* (**sudo**)
- Proces – fundamentalna apstrakcija u UNIX/Linux operacijskim sustavima, koja se definira kao instanca programa u izvršavanju. Ovi sustavi omogućuju paralelno izvršavanje više različitih procesa bilo putem *multiprocessing* ili *multiprogramming* mehanizama.

## 2.2. Arhitektura UNIX/Linux operacijskih sustava

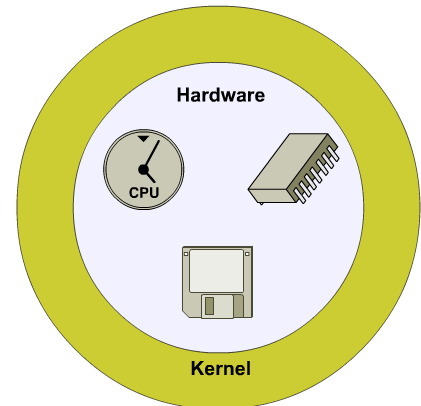
Glavni dijelovi UNIX/Linux operacijskog sustava su:

- **jezgra (engl. kernel)** – upravlja radom računala
- **ljuska (engl. shell)** – interakcija između korisnika i računala (jezgre)
- **datotečni sustav** – organizira i upravlja podacima na vanjskih medijima za pohranu podataka
- **naredbe** – upućuju računalo na obavljanje određenog posla, unose se preko ljuske

### 2.2.1. Jezgra operacijskog sustava

Dio operacijskog sustava najbliži računalnom sklopovlju, tj. izvršna datoteka/e koja se učitava u radnu memoriju računala prilikom pokretanja. Obavlja radnje nužne za rad računala, tri osnovne funkcije:

- upravljanje uređajima (preko upravljačkih programa), memorijom i procesima
- kontroliranje prijenosa informacija između sistemskih programa i sistemskog sklopovlja
- upravljanje servisima, datotečnim sustavom i virtualnom memorijom



Jezgra operacijskog sustava je softver najniže razine koji se pokreće na računalu, što znači da ima mogućnost komunikacije sa sklopovljem. Korisničke aplikacije rade u korisničkom načinu rada i ne mogu izravno pristupiti sklopovlju, ali ni funkcionalnostima operacijskog sustava. Međutim, operacijski sustav zato pruža sučelje za korisničke programe koje definira sve funkcionalnosti jezgre koje takvi programi trebaju, a istovremeno sakriva rad operacijskog sustava. To sučelje se naziva **sustavski pozivi** (engl. *system calls*).

Servisi (engl. *daemons*):

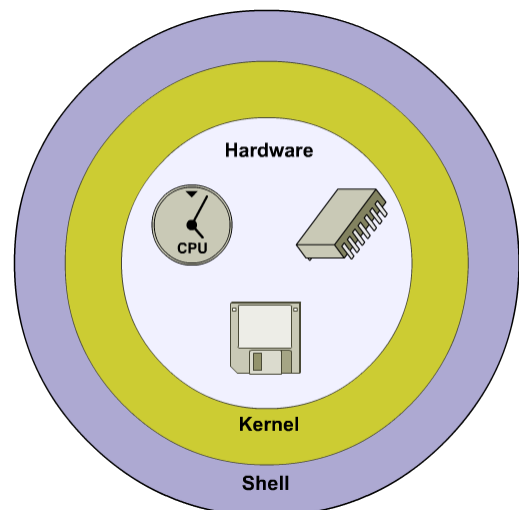
- programi koji obavljanju određeni posao ili nadziru rad jednog dijela sustava
- učitavaju se u memoriju i pokreću prilikom pokretanja OS-a, ostaju aktivni u pozadini dok OS ne zatraži njihovo djelovanje

Virtualna memorija (engl. *swap space*):

- područje (particija) na tvrdome disku rezervirana za virtualno proširenje radne memorije (RAM) računala od strane jezgre
- omogućuje izvršenje većih i većeg broja aplikacija

### 2.2.2. Ljuska operacijskog sustava

Ljuska je sučelje između korisnika i jezgre ili izvršni program koji preuzima ulazne naredbe od korisnika i prenosi ih jezgri na izvršenje (interpreter). Najčešće korištene ljuske su Bourne, Korn i C. Ostale ljuske BASH (Linux default), Z, TC



### 2.2.3. Datotečni sustav

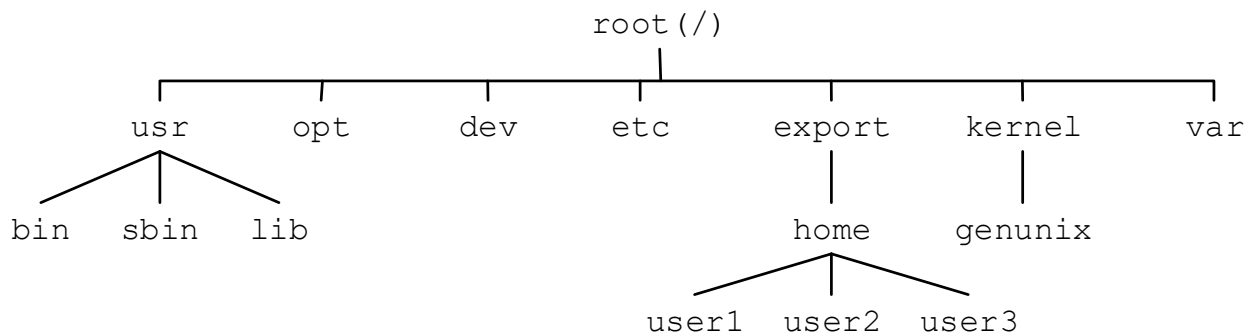
Datotečni sustav razdvaja i prati promjenu informacija na diskovnim sustavima. Definira strukturu datoteka (direktoriji, poddirektoriji i datoteke grupirane skupa iz specifičnih razloga). Podržava integraciju s jezgrom što omogućuje organizacijsku podršku za programe i podatke. Datotečni sustavi se stavljaju na diskovne particije. Moguće je imati više od jedne particije na disku. Tip, svojstva i uređenje datoteka na disku određene su datotečnim sustavom.

Uobičajeni datotečni sustavi:

- Windows: File Allocation Table 32-bit (FAT32), New Technology File System (NTFS), Extended File Allocation Table (exFAT)
- UNIX: UNIX File System (UFS)
- Linux: 3rd Extended File System (ext3), 4th Extended File System (ext4), Global File System (GFS2)...

Hijerarhijska struktura datotečnog sustava operacijskog sustava Linux:

- Korijenski direktorij (engl. *root*) datotečnog sustava je početak datotečnog sustava na disku, označava se sa oznakom '/'
- ispod korijenskog direktorija nalaze se ostali direktoriji
- unutar direktorija se nalaze poddirektoriji (direktorij koji se nalazi unutar drugog direktorija)



Uobičajeni UNIX/Linux direktoriji:

- `/bin` (binary) – sadržava naredbe OS-a
- `/sbin` (single user binaries) – sadrži osnovne izvršne programe za pokretanje i oporavak sustava
- `/boot` (Linux) – sadržava datoteke nužne za pokretanje Linux sustava
- `/kernel` (UNIX) – sadrži osnovni OS (kernel)
- `/usr` (user) – sadržava izvršne naredbe, sistem-administratorske programe i rutine
- `/lib` (library) – sadrži zajedničke knjižnice datoteka
- `/opt` (optional) – sadržava nestandardne programe i programe trećih proizvođača
- `/dev` (devices) – sadržava datoteke koji su pokazivači na uređaje. Sukladno principu „sve je datoteka“
- `/etc` (etcetera) – sadržava većinu konfiguracijskih datoteka sustava
- `/home` (Linux) `/export/home` (UNIX) – sadrži korisničke direktorije
- `/mnt` (mount) – standardno mjesto priključenja datotečnih sustava kod Linux-a
- `/proc` (process) – sadrži datoteke vezane za informacije o sustavu kod Linux-a
- `/var` (variable) – sadrži dinamičke i promjenjive datoteke (print spooling, mail datoteke, log datoteke...)
- `/tmp` (temporary) – sadrži privremene datoteke koje se nakon upotrebe brišu

## 2.3. Mrežni pristup udaljenom Linux poslužitelju

Pristup sustavu može biti lokalni (konzola) ili putem mreže (udaljeni). Također, pristup može biti ostvaren kroz grafičko (engl. *Graphical User Interface* – GUI) ili tekstualno (engl. *Command Line Interface* – CLI) sučelje. Udaljenom poslužitelju se spajamo prema modelu klijent-poslužitelj, koji razlikuje tri tipa računala:

- Domaćin (engl. *Host*) (lokalni ili udaljeni): bilo koje računalo na mreži koje ima TCP/IP protokol
- Poslužitelj (engl. *Server*): daje resurse/servise u mrežu
- Klijent (engl. *Client*): koristi poslužiteljske resurse

Mrežno spajanje na udaljeni poslužitelj može biti ostvareno različitim programima i protokolima. Najpoznatiji protokoli za udaljeni pristup su:

### telnet

(*user interface to the TELNET protocol* – korisničko sučelje za TELNET protokol)

Dio TCP/IP protokola koji simulira ljusku OS koja se pokreće na udaljenom hostu. Potrebno je logirati se kao kod terminalskog pristupa. Iz sigurnosnih razloga zamijenjen ssh protokolom.

### ssh

(*SSH client (remote login program)* – program za udaljeni pristup)

Secure Shell (SSH) ima mogućnosti telnet-a ali uspostavlja sigurnu (kriptiranu) komunikaciju između hostova.

### ftp

(*Internet file transfer program* – program za prijenos podataka)

FTP (File Transfer Protocol) je dio TCP/IP protokola koji omogućuje prijenos podataka u ASCII ili binarnom obliku. Podržava neke naredbe OS-a, za lokalnu izvedbu naredbi ispred naredbe je potrebno dodati slovo 'l'. Najčešće naredbe su put i get.

Kako bi utvrdili uspješnost spajanja, možemo provjeriti dostupnost resursa na mreži, korištenjem naredba **ping** ili **traceroute**.

### 3. UVOD U LJUSKU OPERACIJSKOG SUSTAVA LINUX

Ljuska operacijskog sustava je u suštini program koji prima korisničke naredbe s tipkovnice i šalje ih jezgri operacijskog sustava na izvođenje. Gotovo sve distribucije Linuxa prema zadanim postavkama koriste **Bash (Bourne Again shell)** ljusku. Radno znanje pisanja skripti za ljusku (engl. *shell scripting*) vrlo je bitno je za poslove nalik onom sistemskog administratora, gdje je potrebno brzo i efikasno dohvatiti podatke pohranjene u datotečnom sustavu ili memoriji i to u odgovarajućem formatu.

Za pristup sustavu potrebno je imati korisnički račun na sustavu. Postoje dva osnovna tipa korisničkih računa u Linux operacijskom sustavu:

1. **root** (administratorski) korisnički račun
  - default korisnički račun kreiran prilikom instalacije
  - može pristupiti svim datotekama i svim procesima
  - „Super User“ korisnički račun (kao Administrator na Windows i Admin na Netware OS-u)
  - Administrira ostale korisničke račune
2. Korisnički račun
  - ima pristup vlastitom (home) direktoriju
  - ima ograničen pristup izvršnim programima (`usr/bin`,...) i ostalim datotekama
  - korisnički račun se sastoji od korisničkog imena i zaporka čije kreiranje može biti podložno pravilima koje ovise o sustavu i postavkama administratora
  - podaci korisničkog računa čuvaju se u datoteci `/etc/passwd`. Sadržavaju 7 podataka:
    - korisničko ime, najčešće kombinacija slova imena i prezimena, jedinstveno u sustavu, dodjeljuje ga administrator
    - oznaka polja za zaporku (najčešće znakovi 'x' ili '!'), upućuje da je zaporka kriptirana u `/etc/shadow` datoteci
    - user identification (UID) number – jednoznačna brojana oznaka korisničkog računa
    - group membership (GID) number – oznaka primarne grupe korisnika kojoj pripada korisnički račun
    - korisnički podaci – najčešće korisnikovo puno ime, odjel, broj telefona
    - putanja korisničkog (home) direktorija – lokacija datoteka korisnika u sustavu
    - oznaka ljuske – putanja ljuske koja se pokreće kod pristupa korisnika na sustav

Naredbom **passwd** moguće je promijeniti korisničku zaporku. Administrator ne može pročitati ali može promijeniti bilo koju zaporku.

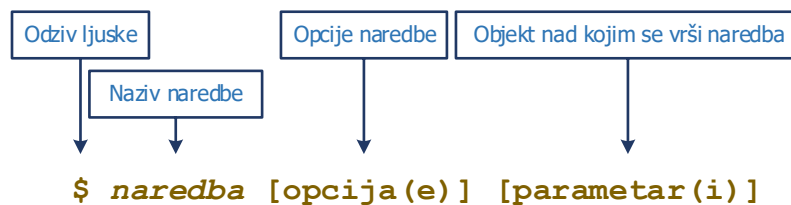
**passwd**

(*change user password* – promjeni korisničku zaporku)

Korisnik prije promjene zaporka iz sigurnosnih razloga mora upisati staru zaporku. Unos nove zaporka mora se potvrditi, a zaporka se prilikom unosa ne ispisuje na zaslonu.

#### 3.1. Osnovne naredbe ljuske

Opća sintaksa naredbi u UNIX/Linux operacijskim sustavima:



Pravila sintakse pisanja naredbi:

- za odvajanje dijelova naredbi koristi se praznina (engl. *space*),
- u jednoj liniji dopušteno je maksimalno 256 znakova,
- naredbe se u pravilu pišu malim slovima,
- opcije mogu biti pisane malim ili velikim slovima (različito značenje) pri čemu uobičajeno započinju s crticom ('-'),
- više opcija može biti korišteno uz jednu crticu (npr. -abcd),
- mnoge naredbe ne zahtijevaju sva tri osnovna dijela naredbe,
- više naredbi može biti uneseno u jedan redak ako se međusobno odvoje točka-zarezom(';')

### 3.2. Jednostavno kretanje kroz datotečni sustav

#### pwd

*(print name of current/working directory – ispisuje naziv trenutnog/radnog direktorija)*

Sve u Linuxu je datoteka, a svaka datoteka je organizirana u hijerarhijsko stablo direktorija. Sintaksa naredbe: **pwd** [-option]. Prvi direktorij u datotečnom sustavu se naziva korijenski (*root*) direktorij. Položaji datoteka i direktorija označavaju se putanjama (*paths*). Kako bi vidjeli trenutni položaj, koristi se naredba **pwd** koja ispisuje apsolutnu putanju do trenutnog (radnog) direktorija.

#### cd

*(change directory – promijeni direktorij)*

Kretanje kroz datotečni sustav temelji se na putanjama. Sintaksa naredbe: **cd** [-option] [pathname]. Postoje 2 načina za specificiranje putanje:

- Apsolutna putanja – putanja od korijenskog direktorija ('/')
- Relativna putanja – putanja od trenutnog direktorija

Kako bi promijenili direktorij u kojem se nalazimo, koristi se naredba **cd** koja kao parametar prihvaća i apsolutnu i relativnu putanju. Navigacija pomoću apsolutnih ili relativnih putanja može biti ubrzana korištenjem prečaca:

- **cd .** → trenutni direktorij
- **cd ..** → roditeljski direktorij
- **cd ~** → *home* direktorij

#### ls

*(list directory contents – izlistaj sadržaj direktorija)*

Naredba **ls** izlistava sve direktorije i datoteke u trenutnom direktoriju ili u određenom direktoriju koji je naveden kao parametar naredbe. Sintaksa naredbe: **ls** [-option(s)] [pathname(s)]. Bez navođenja opcija daje 'široki' ispis bez mogućnosti raspoznavanja



## touch

(change file timestamps - promjena vremenskog obilježja datoteke)

direktorija i datoteka. Razvrstava ispis po abecednom redu (specijalni znakovi, brojevi, velika slova, mala slova). Datoteke čije ime započinje s . su skrivene i neće biti prikazane osnovnom uporabom naredbe *ls*. Najčešće opcije uz naredbu su:

- *-a = all*; izlistava i skrivene datoteke
- *-l = long*; detaljna lista datoteka u dugačkom formatu (dozvole datoteka, broj linkova, ime vlasnika, grupa vlasnika, veličina datoteke, vremenska oznaka zadnje izmjene i naziv datoteke/direktorija)
- *-ld = list directory*; izlistava direktorij bez njegova sadržaja
- *-R = recursive*; izlistava sadržaj direktorija i svih poddirektorija
- *-F = file classify*; za prikaz tipova datoteka

U većini Linux distribucija, naredba **ls** generira obojani ispis prema zadanim postavkama. Tablica 3 prikazuje sheme boja za sve vrste datoteka unutar operacijskog sustava Linux.

**Tablica 3.** Zadane sheme boja uz naredbu *ls* unutar operacijskog sustava Linux

Vrsta datoteke	Boja
<b>direktorij</b>	plava
<b>kompresirana arhiva</b>	crvena
<b>tekstualne datoteke</b>	bijela
<b>slike</b>	roza
<b>link</b>	cyan
<b>uređaj</b>	žuta
<b>izvršna datoteka</b>	zelena
<b>pokvaren link</b>	treptuće crvena

### 3.3. Jednostavan rad s datotekama

Koristi se za izradu nove, prazne datoteke. Sintaksa naredbe: **touch** [-option(s)] [-filename]. Ako datoteka *filename* postoji **touch** samo osvježava datum/vrijeme modifikacije te datoteke. Korisnik mora imati ovlasti pisanja na dijelu datotečnog sustava gdje izrađuje datoteku.

## file

(determine file type – određuje tip datoteke)

Koristi se za određivanje tipa datoteke. Sintaksa naredbe: **file** [-option(s)] [-filename]. U Linuxu nazivi datoteka ne moraju predstavljati sadržaj datoteke. Izlaz je najčešće:

- ASCII Tekst
- Izvršna datoteka (naredba, program ili skripta)
- Podaci (datoteka kreirana nekom aplikacijom)

## cat

(concatenate files and print on the standard output – ulančava datoteke i ispisuje ih na standardni izlaz)

Koristi se za ispis cijelog sadržaja ASCII datoteka. Sintaksa naredbe: **cat** [-option(s)] [-filenames]. Uglavnom se koristi za ispis manjih datoteka, kod velikih datoteka se ispis može

kontrolirati sa Ctrl+S/Ctrl+Q – nepraktično kod brzih računala. Često se koristi za 'ulančavanje' (povezivanje) više datoteka kraćeg sadržaja.

#### **more**

*(file perusal filter for crt viewing – preglednik datoteka za monitor)*

Prikazuje sadržaj tekstualne datoteke ekran po ekran. Sintaksa naredbe: **touch** [-option(s)] [-filename]. Pritiskom na tipku **h** dobiva se sustav pomoći za navigaciju kroz preglednik (koristi se za pregled **man** stranica). Naredba **less** omogućuje sličnu ali napredniju funkcionalnost.

#### **cp**

*(copy files and directories – kopiraj datoteke i direktorije)*

Koristi se za kopiranje datoteke(a) na novu lokaciju, može kopirati i na novu lokaciju unutar istog ili različitog direktorija. Sintaksa naredbe: **cp** [source\_filename] [destination\_path]. Ne mogu postojati dvije datoteke s istim imenom unutar istog direktorija. Ukoliko odredišna putanja nije dobro definirana, naredba javlja grešku. izvorišne datoteke zadržavaju svoj naziv i lokaciju dok se eksplicitno ne preimenuju:

- **-i** opcija sprječava slučajno prepisivanje datoteka (ako odredišna datoteka postoji traži potvrdu kopiranja/prepisivanja postojeće datoteke)
- **-r** opcija se koristi za kopiranje direktorija zajedno s njegovim sadržajem
  - sintaksa: **cp -r[i]** [source\_filename] [destination\_path].

#### **mv**

*(move (rename) files – premjesti (preimenuj) datoteke)*

Datoteke mogu biti preimenovane i premještene korištenjem naredbe **mv**. Preimenovanje datoteka u tekućem direktoriju mijenja ime izvorišne datoteke, a ne kopira samu datoteku. Sintaksa naredbe: **mv** [-i] [source\_filename] [destination\_path]. Kod premještanja datoteke(a) u drugi direktorij, nakon izvođenja naredbe izvorišna datoteka(e) postoji samo u odredišnom direktoriju. Navođenjem odredišnog direktorija i novog imena datoteke naredbom **mv** može se istovremeno i premjestiti i preimenovati datoteka.

Naredba također služi i za premještanje direktorija i njegovog sadržaja, a ako odredišni direktorij postoji izvorišni direktorij se samo kopira. Ako odredišni direktorij ne postoji izvorišni direktorij se samo preimenuje.

#### **mkdir**

*(make directories – kreiraj direktorije)*

Koristi se za izradu novog direktorija. Sintaksa naredbe: **mkdir** [-p] [directory\_name]. Opcija **-p** (engl. *parent*) automatski pravi sve direktorije u putanji koji nedostaju.

#### **rm**

*(remove files or directories – ukloni datoteke ili direktorije)*

Koristi se za trajno uklanjanje jedne ili više datoteka iz datotečnog sustava. Sintaksa naredbe: **rm** [-i] [filename(a)]. Za brisanje većeg broja datoteka često se nazivi datoteka kombiniraju s metaznakovima:

- **-i** (engl. *interactive*) opcija omogućuje upit prije trajnog brisanja datoteke
- **-r** (engl. *recursive*) opcija koristi se za brisanje direktorija zajedno s njegovim sadržajem
  - sintaksa: **rm -r[i]** ime\_direktorija

## man

(an interface to the on-line reference manuals – sučelje prema on-line priručnicima)

Za brisanje praznog direktorija može se koristiti naredba **rmdir**.

### 3.4. Naredbe za olakšano korištenje znakovnog sučelja

## history

(GNU History Library – povijest korištenih naredbi ljske)

U ljsuci se nalazi povijest naredbi koje su prethodno unesene te se može pregledati. Ova naredba je vrlo korisna kada želite pronaći i pokrenuti naredbu koju ste prethodno koristili, a da ju ne upisujete ponovno. Želite li pokrenuti istu naredbu koju ste unijeli prije, pritisnite strelicu za gore na tipkovnici. Ako želite pokrenuti prethodnu naredbu bez da ju ponovno upišete, koristite **!!**. Drugi prečac za otkrivanje povijesti je kombinacija tipaka **Ctrl+R**, nakon koje možete početi tipkati dijelove naredbu koju želite te će ljsuka prikazati podudarnosti.

## clear

(clears the terminal screen – briše zaslon terminala)

Koristi se za uklanjanje trenutnog sadržaja na terminalu, u svrhu čistijeg prikaza.

U kontekstu olakšanog korištenja znakovnog sučelja ili terminala, jedna od najkorisnijih značajki je dovršavanje rečenice (engl. *tab completion*). Ako započnete tipkati početak naredbe, datoteke, mape i sl., pritisnite tipku **Tab** i ona će se automatski dovršiti na temelju onoga što pronađe u direktoriju u kojem se nalazite. Popis svih kombinacija tipki koje olakšavaju rad u Bash ljsuci su prikazane u Tablici 4.

**Tablica 4.** Kombinacije tipki u Bash ljsuci

Kombinacija tipki	Funkcionalnost
<b>Ctrl+A</b>	Pomicanje kursora na početak linije komandnog sučelja
<b>Ctrl+C</b>	Završetak pokrenutog programa
<b>Ctrl+D</b>	Odjavljivanje iz trenutne sesije ljsuke
<b>Ctrl+E</b>	Pomicanje kursora na kraj linije komandnog sučelja
<b>Ctrl+L</b>	Čišćenje terminala
<b>Ctrl+R</b>	Pretraživanje povijesti komandnog sučelja
<b>Ctrl+Z</b>	Suspendiranje procesa
<b>Strelica gore/dolje</b>	Pretraživanje povijesti naredbi
<b>Tab</b>	Završetak naziva naredbe ili datoteke

### 3.5. Naredbe za pretraživanje Linux priručnika

**man** stranice se službeno nazivaju „UNIX Programmers's Manual“. Pružaju informacije o naredbama, sustavskim pozivima, formatima datoteka i održavanju sustava. Standardni su dio svih UNIX/Linux sustava, tekstualne, pokreću se iz CLI. Dijelovi man stranica su:

- **NAME** - naziv naredbe i nazivi sličnih naredbi
- **SYNOPSIS**- prikazuje sintaksu naredbe i raspoložive opcije i parametre
- **DESCRIPTION** - pregled djelovanja naredbe

## whatis

*(display manual page descriptions – ispisuje opis djelovanja naredbe)*

- OPTIONS - raspoložive opcije koje mijenjaju funkciju ili efekt naredbe
- OPERANDS - objekti naredbe na kojima se ona izvršava
- SEE ALSO - upućuje na povezane naredbe i teme

Naredba **apropos** se koristi za pretraživanje rječnika prema ključnoj riječi. Ima jednako značenje kao i naredba **man -k keyword**.

Nemaju sve man stranice sve dijelove, dok neke imaju i više. Sve naredbe imaju minimalno NAME, SYNOPSIS i DESCRIPTION dijelove.

Koristi se kada korisnik poznaje naziv naredbe ali ne zna čemu naredba služi.

## 4. ZADACI ZA SAMOSTALNI RAD

Pristupiti poslužitelju linux.etfos.hr kako slijedi:

1. http pristup:

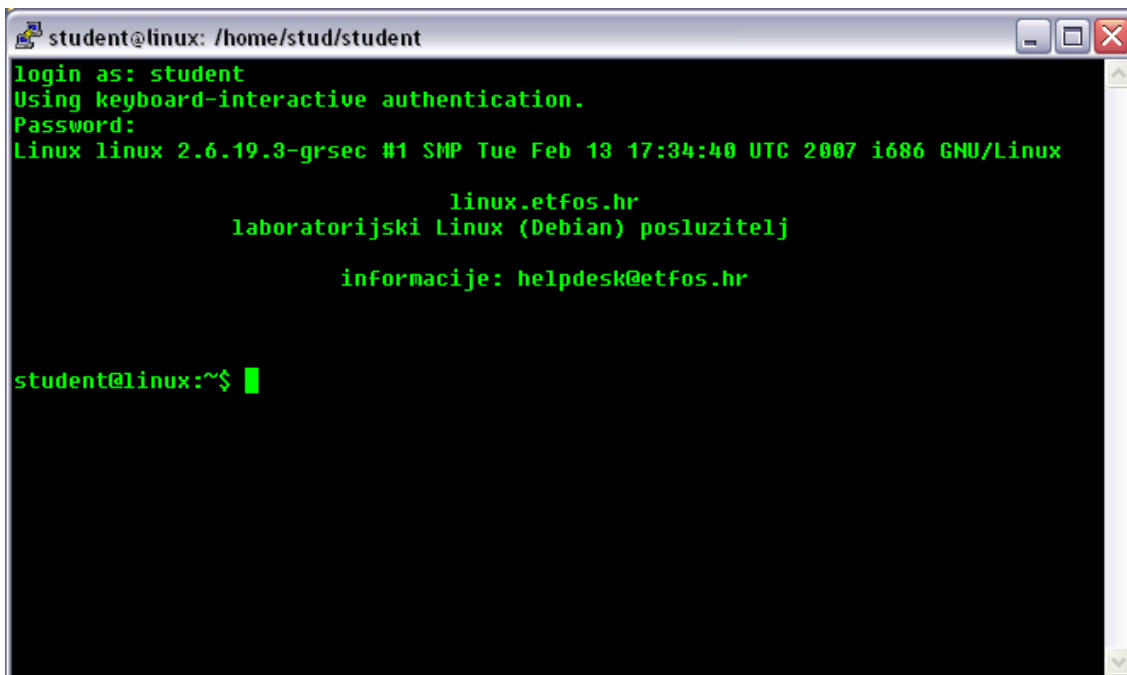
Internet Explorer/Mozilla Firefox,...: <http://linux.etfos.hr/>

- nastavni materijali, programi,...

2. SSH pristup:

ssh klijent (ssh, Putty,..): linux.etfos.hr

(download: <http://linux.etfos.hr/alati/putty.exe>)



```
student@linux: /home/stud/student
login as: student
Using keyboard-interactive authentication.
Password:
Linux linux 2.6.19.3-grsec #1 SMP Tue Feb 13 17:34:40 UTC 2007 i686 GNU/Linux

                    linux.etfos.hr
      laboratorijski Linux (Debian) poslužitelj

      informacije: helpdesk@etfos.hr

student@linux:~$
```

Odgovorite na sljedeća pitanja:

1. Odrediti podatke korisničkog računa s kojim ste se spojili na linux.etfos.hr:  
korisničko ime: \_\_\_\_\_  
oznaka polja za zaporku (najčešće znakovi 'x' ili '!'): \_\_\_\_\_  
user identification (UID) number: \_\_\_\_\_  
group membership (GID) number: \_\_\_\_\_  
korisnički podaci: \_\_\_\_\_  
putanja korisničkog (home) direktorija: \_\_\_\_\_  
oznaka ljuske: \_\_\_\_\_
2. Pozicionirajte se u */proc* direktorij. Proučite sadržaj direktorija i pronađite datoteku u kojoj se nalaze sljedeće informacije:
  - a. Koje particije čine datotečni sustav poslužitelja?
  - b. Koliko vremena poslužitelj radi (*uptime*)?
  - c. Koji procesor se nalazi u poslužitelju? Navedite ime modela, frekvenciju takta i broj jezgri.

3. Korištenjem naučenih naredbi za jednostavno kretanje kroz datotečni sustav, odgovorite na sljedeća pitanja:
- Na koji način možete odrediti direktorij u kojem se nalazite?
  - Pokrenite naredbu **cd** bez dodatnih opcija i zaključite gdje vas je to odvelo. Ako se trenutno nalazimo u direktoriju `/home/student50/dir1` i želimo otići u direktorij `/home/student50`, koji prečac možemo koristiti?
  - Koristeći relativnu putanju izlistati u dugom obliku sadržaj direktorija `dir1`. Navesti izlistane direktorije i izlistane datoteke. Koje su skrivene?
  - Pokrenite naredbu **ls** s opcijama `-R`, `-r` i `-t` i opišite dobiveni izlaz. Koju naredbu možemo koristiti za prikaz skrivenih datoteka?
4. Korištenjem naučenih naredbi za jednostavan rad s datotekama, odgovorite na sljedeća pitanja:
- Pokrenite naredbu **file** za nekoliko različitih direktorija i datoteka i opišite dobiveni izlaz.
  - Pokrenite naredbu **cat** na različitim datotekama i direktorijima. Zatim pokušajte spojiti više datoteka u naredbi. Opišite dobivene izlaze.
  - Kreirajte novu datoteku oblika `<ime.prezime>` u vašem home (`cd ~`) direktoriju. Preimenujte datoteku u oblik `<prezime.ime>`.
  - Kreirajte novi direktorij u vašem home direktoriju. Možete li premjestiti vaš novi direktorij na razinu na kojoj je i `/home` direktorij? Kopirajte cijeli sadržaj iz direktorija `dir1` u vaš novi direktorij. Izlistajte sadržaj novog direktorija sortiran po obrnutom abecednom redoslijedu.
5. Korištenjem naučenih naredbi za jednostavno pretraživanje rječnika, odgovorite na sljedeća pitanja:
- Proučiti naredbu **passwd** u prvoj i petoj sekciji. Koje područje upotrebe naredbe **passwd** opisuje pojedine sekcije?
  - Izlistati sve naredbe koje u sebi sadržavaju pojam `ssh`. Koja su dva načina da se to učini?
  - Koje opcije **ls** naredbe omogućuju sortiranje ispisa sadržaja direktorija prema vremenu modifikacije datoteke te prema veličini datoteke?
  - Pomoću naredbe **man** proučiti dodatne mogućnosti naredbi **cp** i **mv** te navesti barem 3 dodatne mogućnosti za svaku.

## 5. LITERATURA

- [1] Stallings, W., 2012. Operating systems: internals and design principles. Boston: Prentice Hall,.
- [2] Tanenbaum, A.S. and Bos, H., 2015. Modern operating systems. Pearson.
- [3] Bovet, D.P. and Cesati, M., 2005. Understanding the Linux Kernel: from I/O ports to process management. " O'Reilly Media, Inc."
- [4] Love, R., 2010. Linux kernel development. Pearson Education.
- [5] Love, R., 2013. Linux system programming: talking directly to the kernel and C library. " O'Reilly Media, Inc."
- [6] Jelenković L., 2019. Operacijski sustavi: Interni material za predavanja iz predmeta. Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva ([http://www.zemris.fer.hr/~leonardo/os/fer/\\_OS-skripta.pdf](http://www.zemris.fer.hr/~leonardo/os/fer/_OS-skripta.pdf))
- [7] <https://linuxjourney.com/>
- [8] <https://www.geeksforgeeks.org/operating-systems/>
- [9] <https://labex.io/courses/linux-for-noobs>
- [10] <https://labex.io/courses/linux-basic-commands-practice-online>
- [11] [https://www.linuxtopia.org/online\\_books/introduction\\_to\\_linux/sect\\_02\\_05.html](https://www.linuxtopia.org/online_books/introduction_to_linux/sect_02_05.html)
- [12] <https://tldp.org/>