Uvod v Bash Shell programiranje v Linux

dr. Biljana Mileva-Boshkoska

Fakulteta za informaciske študije

July 2014, Novo mesto, Slovenia

Univerza v Ljubljani
Fakulteta za strojništvo

Fakulteta za informacijske študije
Faculty of information studies

Kreativno jedro:
Simulacije
Creative core: Simulations

Naložba v vašo prihodnost
OPERACIJO DELNO FINANCIRA EVROPSKA UNIJA
Evropski sklad za regionalni razvoj

REPUBLIKA SLOVENIJA
MINISTRSTVO ZA IZOBRAŽEVANJE, ZNANOST IN ŠPORT

Kazalo

Kaj je operacijski sistem?

- Operacijski sistem (OS) je programska oprema, ki upravlja s strojno opremo.
- Glavne naloge:
  - Vmesnik med uporabniškim nivojem in računalniškimi viri
  - Interakcija in komunikacija z uporabniki
  - Upravljanje z resursi
  - Nabor koristnih uslužnostnih rutin
  - Množica pomagal za razvoj in upravljanje s projekti

Osnovne funkcionalnosti:

- Upravljanje s procesi
- Upravljanje z napravami
- Upravljanje s pomnilnikom
- Upravljanje z zbirčnimi sistemi
- Upravljanje z omrežjem

Uvod v operacijski sistem Linux

- Linux je varianta operacijskega sistema UNIX
- Unix je ena od najstarejših vrst operacijskih sistemov, ki zagotavlja zanesljivost in varnost v profesionalnih programih že skoraj pol stoletja.
- Veliko strežnikov po svetu, ki shranjujejo podatke za priljubljena spletišča (kot sta YouTube in Google) poganjajo različice sistema Unix.
- Unix je bil popolnoma osnovan na vmesniku ukazne vrstice do začetka 1990, ko so se začeli pojavljati grafični uporabniški vmesniki.
- Odprtokodna in brezplačna sistemska programska oprema

Zgodovinski razvoj Linuxa

- Linux 0.01, 1991: 9300 vrstic kode v C in 950 vrstic kode v asemblerju, Linus Torvalds - prva verzija, baziran na veliko idej iz MINIX sistema (tudi UNIX sistem), virtualni spomin, bolj sofisticiran zbirčni sistem ...
- Linux 1.0, 1994: 176.250 vrstic kode v C-ju, novi zbirčni sistem, spominsko preslikane datoteke, TCP/IP protokoli, gonilniki ...
- Linux 2.0, 1996: 470 000 vrstic kode v C-ju, 8000 vrstic kode v asemblerju, podpora 64-bitnih arhitektur, simetrično multiprogramiranje, novi omrežni protokoli ...
- Linux 2.2.0, 1999: 1,800,847 vrstic kode ...
- Linux 2.4.0, 2001: 3.377.902 vrstic kode ...
- Linux 2.6.0, 2003: 5,929,913 vrstic kode ...

Linux jedro

- Jedro Linux je nadzornik operacijskega sistema.
- Odgovorno je za dodeljevanje pomnilnika in procesorskega časa.
- Linux jedro - program, ki upravlja z vsemi programi na računalniku.
- Linux jedro - operacijski sistemi, ki temeljijo na Linux jedra, so običajno v obliki distribucij Linuxa.

Linux distribucije

- Distribucija vključuje veliko kolekcijo raznorodne odprtokodne in brezplačne programske opreme ki je organizirana v obliki paketov.
- Trenutno obstaja več kot 600 različnih Linux distribucij
- Popularne distribucije
  - Debian – nekomercialna distribucija
  - Ubuntu – popularna namizna in serverska distribucija, jo bomo uporabljali na delavnici
  - Fedora – distribucija, sponzorirana od ameriškega podjetja Red Hat
  - Gentoo – distribucija, namenjena bolj zahtevnim uporabnikom
  - Redhat – distribucija, namenjena bolj zahtevnim uporabnikom
  - CentOS – distribucija, različica Redhata, ki je na FIŠevem HPC

Ubuntu Linux

- Na delavnici bomo uporabljali Ubuntu 12.04 LTS verzija jedra 3.11.0.17 generic
- Več na spletu: `www.ubuntu.com` in `www.ubuntu.si` (slovenska različica)
- Pomožno gradivo o Ubuntu linuxu v slovenščini `https://www.ubuntu.si/wordpress/wp-content/uploads/ubuntu-manual-sl-final-1204.pdf`
- Več informacij `https://help.ubuntu.com`

Namestitev Ubuntu Linuxa

- Na računalnikih v predavalnici bomo najprej namestili operacijski sistem kot navidezni stroj
- VMware programska oprema za virtualizacijo je že nameščena na računalnikih v predavalnici
- Če želite namestiti VMware na svojem računalniku, je programska oprema dosegljiva na URL-ju `http://www.vmware.com/products/player`
- V VMware zberite "Create a new virtual machine"
- Kot "Installer disc image file (iso)" izberite ISO datoteko, ki je v C:Ubuntu imeniku

Prednosti uporabe Shell

1. Pod Linuxom obstaja nekaj močnih orodij, ki niso na voljo v operacijskem sistemu Windows
2. Eden od teh orodij je nekaj, kar se imenuje "shell programiranje".
3. Najpogostejši Linux shell je "Bash".
4. Uporabite Bash shell, saj se s tem poveča možnost, da bodo vaše skripte prenosni med stroji, delitvami, celo operacijskih sistemov.
5. Shell programiranje je umetnost
6. Ker lupina programiranje je umetnost, prosim, ne pisati reči, "Vau, to je bilo res neučinkovit način, da to tako-in-tako."

Q & A

1. Q. Kakšen je najboljši način, da ugotovim, kateri shell uporabljam?
2. A1. echo $SHELL
3. A2. ps p $$

Uvod v Shell

1. Obstajata dva načini uporabe Shell-a: interaktivno in s pisanjem skripte.
   - V interaktivnem načinu, uporabnik vnese sam ukaz (ali kratek niz ukazov) in rezultat je natisnjen.
   - V shell skripte, uporabnik vnese karkoli - od nekaj vrstic do celotnega programa v urejevalnik besedila, potem izvede nastalo besedilno datoteko kot skripte.
2. Pogosto se dogaja, da interaktivno sejo postane shell skripta, ko se stvari preveč zapletena za enostavne vnose interaktivni linije, ali zato, ker se zdi, da je splošno koristno in vredno ohraniti posebno zaporedje.

Uvod v Shell (2)

3. V sodobnem okolju Linux uporabnik ima lahko več kot eno lupino odprto istočasno, bodisi s premikanjem med zaporedjem neodvisnih "virtualnih terminalih" v samo besedilo okolju, ali tako, da odprete poljubno število navideznih oken v X Windows okolje.
4. Prednost imajo več kot en lupina je na voljo, da bi se ena lupina lahko uporabljajo za testiranje en ukaz naenkrat, medtem ko bi drugi zagotavljajo urejevalnik besedila za sestavljanje posamezne ukaze v programu Shell.
5. ne želim, da bi dobili preveč distribucije specifična, ampak, če niste gostovanje X Windows in želijo več kot eno hkratno shell sejo, s številnimi trenutnih distribucijah lahko preklapljate med "virtualnih terminalov" s pritiskom na Ctrl + Alt + F (n), n običajno med 1 in 6.
6. V okolju, ki podpira X Windows, preprosto odprite poljubno število ukaznih lupinah oken in premikanje med njimi.

Osnovne ukaze v Linux lupini (shell)

- Osnovni Linux shell ukazi
  - Ukaz za spreminjanje imenika (cd)
  - Ukaz za listanje datotek (ls)
  - Dovoljenja za pristop datoteki
  - Ukaz za tip datoteke (file)
  - Ukaz za listanje vsebine datoteke (less, more)
  - Absolutna in relativna pot

Ukazi

- Konvencija pisanja:
- $ date
- Odgovor računalnika:
- Tue Dec 23 10:52:51 PST 2003
- Shell prompt $
- Sintaksa ukazov

```
$ime_ukaza -opcije [argumenti]
```

- Vsak ukaz je sestavljen iz:
  - imena ukaza (imena z velikimi in malimi črkami se razlikujejo)
  - opcij ali argumentov v formatu $-x$, x je neki znak (POMEMBNO: med $-$ in x ni presledka)
  - argumenti so lahko imena datotek, imenik ali izrazi, odvisno od ukaza

Primer ukaza

- Primer:

```
$ ls -a Downloads
```

ls je ukaz za listanje vsebine imenika,
–a je opcija za prikaz seznama vseh datotek;
Downloads je argument.

- En ukaz ima lahko več opcij, ki opredeljujejo, kako se bo ukaz izvrševal.

```
$ ls -a -l Downloads
```

- Opcij se lahko združijo

```
$ ls -al Downloads
```

Primeri ukazov

- Pomoč

  ```
  $ man ime_ukaza
  ```

  Primer:

  ```
  $ man ls
  ```

- Ukaz za prikaz trenutnega datuma in časa

  ```
  $ date
  ```

- Ukaz za prikaz koledarja

  ```
  $ cal
  ```

- Ukaz za konec trenutne seanse (ang. session)

  ```
  $ exit
  ```

Delo z imeniki

- V Linuxu obstaja hierarhična struktura imenikov (ang. directorium), v katerih so organizirane datoteke.
- Na vrhu strukture je poseben imenik, ki se imenuje koren (ang. root).
  $ pwd
  /path/path/path
- Imenik je datoteka, ki vsebuje seznam drugih datotek in ne vsebuje drugih podatkov.
- Vsak uporabnik sistema ima svoj imenik (home directory). Kako do svojega imenika?
  $ cd ~
  $ pwd (Ukaz za prikazovanje trenutnega imenika)
  /home/username
- Različica: $ cd /home/username
- Pri logiranju v sistem postaja trenutni imenik home directory.

Absolutna pot

- Ukaz za spremembo trenutnega imenika

  ```
  $ cd pot_do_imenika
  ```

- Absolutna pot se začne z root imenikom, ki mu sledi seznam imenikov po strukturi do želenega imenika.
- Primer: absolutna pot do imenika, ki vsebuje sistemske programe je:

  ```
  /usr/bin
  ```

  Mu lahko pristopimo z ukazom:

  ```
  $ cd /usr/bin
  ```

Relativna pot

- Relativna pot se začne od trenutnega imenika, ki ga beležimo z simbolom . (pika)
- Starševski imenik se beleži z .. (dvojna pika)
- Primer: iz imenika */usr/bin* hočemo priti v */usr* z uporabo relativne poti

```
$ cd ..
```

- Primer: iz imenika */usr* hočemo iti v */usr/bin* z uporabo relativne poti

```
$ cd ./bin
```

Datoteke

- V imenu razlikujemo med malimi in velikimi črkami; je dovoljena raba številk in nekaterih posebnih znakov, kot so _ # @
- V imenu je prepovedana raba: presledka, metaznakov * ? < > | / ; & ! [ ] $ ' ", ne sme se začeti na + in -
- Datoteke, katerih imena se začnejo z znakom .(pika), so skrite.
- Ukaz $ ls prikazuje vse datoteke iz trenutnega imenika (brez skritih datotek)
- Skrite datoteke se vidijo z ukazom:

```
$ ls -a
```

Naloga

Poskusite in razmislite kaj dela naslednji ukaz:
$ ls -ltr
Navodilo: Lahko preverite opcij ukaza z: $man ls

Kaj dela ukaz ls -l?

1. Vrsta datoteke in dovoljenje za pristop do datoteke
2. Število povezav do datoteke
3. Lastnik datoteke
4. Velikost datoteke v bajtih
5. Čas zadnje spremembe datoteke
6. Ime datoteke

Vrsta datoteke in dovoljenje za pristop

- Vrsta datoteke
  - – navadna datoteka
  - d imenik
  - l povezava
- Uporabniki datotek
  - user – pravice pristopa lastnika datoteke
  - group – pravice pristopa uporabnikov, ki so člani iste skupine kot lastnik datoteke
  - other – pravice pristopa datoteke za vse ostale uporabnike
- Različna dovoljenja za pristop
  - r – dovoljenje za branje
  - w – dovoljenje za zapis
  - x – dovoljenje za izvrševanje

Pomen dovoljenja za datoteke in imenike

- r – dovoljenje za branje
  - dovoljenje za branje datoteke
  - dovoljenje za izpis vsebine imenika
- w – dovoljenje za zapis
  - dovoljenje za spremembo vsebine datoteke
  - dovoljenje za spremembo vsebine imenika (dodajanje in brisanje podimenikov in datotek)
- x – dovoljenje za izvrševanje
  - dovoljenje za izvrševanje datoteke
  - dovoljenje imenika, naj postane trenuten imenik

Ukaz chmod

- Se uporablja za spremembo dovoljenj datoteke ali imenika.
- Spremembo lahko naredi samo lastnik datoteke in tako imenovani nadouporabnik (ang. superuser, poglete ukaz su)
- Podpira dva različna načina spremembe dovoljenja: z uporabo oktalnih številk in z uporabo simbolov

Simbolna notacija

- Zaenkrat bomo fokusirani na uporabo simbolov za spremembo dovoljenja. Notacija je razdeljena v 3 sklope:
  - na koga se sprememba nanaša
    - u (uporabnik, lastnik)
    - g (skupina)
    - o (vsi ostali)
    - a (vsi kombinacija u g o)
  - katera operacija se bo izvršila
    - + (dodaj dovoljenje)
    - - (umakni dovoljenje)
    - = (samo posebna dovoljenja bodo izvršena, vsa ostala bodo umaknjena)
  - katero dovoljenje bo nastavljeno
    - r (branje)
    - w (zapis)
    - x (izvrševanje)

Primeri simbolne notacije za ukaz chmod

Notacija	Pomen
u+x	Dodaj dovoljenje za izvrševanje uporabniku
u-x	Umakni dovoljenje za izvrševanje uporabniku
+x	Dodaj dovoljenje za izvrševanje uporabniku, skupini in vsem ostalim. Ekvivalentno notaciji a+x
o-rw	Umakni dovoljenja za branje in pisanje vsakomur, razen lastniku in lastniku skupine

Primer: $chmod +x ime_datoteke.sh

Ukaz za tip datoteke

- `$ file ime_datoteke`
- Primer:

```
$ file picture.jpg

picture.jpg: JPEG image data
$ file tux_small.png
tux_small.png: PNG image data, 128 x 151, 8-bit/color RGB,
non-interlaced
```

Ukaza less in more

- Se uporablja za izpis vsebine tekstualne datoteke
- `$ less ime_datoteke`
- `$ more ime_datoteke`
- Primer:

```
$ less /etc/passwd

$ more /etc/passwd
```

Zakaj ukazi namesto grafični vmesnik?

- Moč in prilagodljivost
- Kompleksne naloge se izvršujejo hitreje kot v grafičnem vmesniku
- Primer: Na kakšen način bi kopirali vse HTML datoteke iz enega imenika v drugi, ampak samo tiste, ki ne obstajajo že v ciljnem imeniku ali so novejše od verzij datotek v ciljnem imeniku?

```
$ cp -u *.html destination
```

Upravljanje z datotekami in imeniki

- Ukazi za upravljanje z datotekami in imeniki
  - Kopiranje datotek in imenikov (*cp*)
  - Premikanje ter preimenovanje datotek in imenikov (*mv*)
  - Ustvarjanje imenikov (*mkdir*)
  - Brisanje datotek in imenikov (*rm*)

Ukaz mkdir - make directorium

- Se uporablja za ustvarjanje imenika
- `$mkdir ime_imenika...`
- Primer – ukaz za ustvarjanje enga imenika z imenom ime1

  `$ mkdir ime1`
- Primer – ukaz za ustvarjanje treh imenikov z imeni ime1, ime2, in ime3

  `$mkdir ime1 ime2 ime3`
- Ustvarjeni imeniki so podimeniki trenutnega imenika

Ukaz cp - copy

- Se uporablja za kopiranje datotek in imenikov
- `$ cp origin destination`
- Primer: kopiranje datoteko stvar 1 v datoteko stvar 2

  `$ cp stvar1 stvar2`
- Kopiranje več stvari (datotek ali imenikov) v drugi imenik

  `$ cp stvar... imenik`
- Poglejte opcijo -R ($cp -R)

Opcije ukaza cp

Opcija	Pomen
–a	Kopiraj datoteke in imenike vključno z vsemi atributi in dovoljenji. Kopije dobijo atribute in dovoljenja uporabnika, ki izvaja kopiranje.
–i	Pred prepisom (ang. overwrite) že obstoječe datoteke, pozovi uporabnika, naj potrdi.
–r	Rekurzivno kopiraj imenike in njihovo vsebino. Opcija se uporablja pri kopiranju imenikov, če hočemo, da se kopirajo tudi vse datoteke in podimeniki imenika, ki ga kopiramo.
–u	Pri kopiranju datotek iz enega imenika v drugega kopiraj samo datoteke, ki ne obstajajo ali so novejše od že obstoječih datotek v ciljnem imeniku.
–v	Prikaži informativna sporočila, medtem ko se izvaja kopiranje.

Ukaz mv - move

- Se uporablja za premikanje ter preimenovanje datotek in imenikov
- primer: premikanje ali datoteke "prva" v datoteko "druga"

```
$ mv prva druga
```

Opcije ukaza mv

Opcija	Pomen
–i	Pred prepisom (ang. overwrite) že obstoječe datoteke, pozovi uporabnika, naj potrdi.
–u	Pri premikanju datotek iz enega imenika v drugega premakni samo datoteke, ki ne obstajajo ali so novejše od že obstoječih datotek v ciljnem imeniku.
–v	Prikaži informativna sporočila, medtem ko se izvaja premikanje.

Ukaz za brisanje datotek in imenikov (rm)

- Se uporablja za brisanje ene ali več datotek in imenikov

```
$ rm stvar...
```

- Pozor: rm * izbriše vse v tekovnem imeniku

Opcije ukaza rm

Opcija	Pomen
–i	Pred brisanjem že obstoječe datoteke, pozovi uporabnika, naj potrdi. Če ta opcija ni navedena, bo ukaz potiho pobrisal datoteko.
–r	Rekurzivno pobriši imenike in njihovo vsebino. Opcija se uporablja pri brišenju imenikov, če hočemo, da se brišejo tudi vse datoteke in podimeniki imenika, ki ga brišemo.
–f	Ne upoštevaj neobstoječe datoteke in ne pozivaj uporabnika, naj potrjuje. To razveljavi opcijo –i.
–v	Prikaži informativna sporočila, medtem ko se izvaja brisanje.

Kaj je shell skripta?

- Shell skripta je datoteka ki vsebuje niz ukazov.
- Shell prebere datoteko in izvaja ukaze kot da bi bili vnešeni direktno na ukazni vrstici.
- močnen vmesnik ukazne vrstice do sistema in interpreter skriptnega jezika hkrati
- avtomatizacija dela v shell okolju in poenostavljenje nalog
- SH in BASH sta dve najbolj uporabljani shell okolini

Kako pišemo shell skripte?

1. Napiši skripto
   - shell skripte so navadne tekstualne datoteke
   - pišejo se z uporabo urejevalnika tekstov
   - večino urejevalnikov uporablja skladenjsko označevanje (ang. syntax highlighting)
   - datoteka se shranjuje z ekstenzijo .sh (primer skripta.sh)
2. Naredi da je skripta izvršljiva
   - navadne tekstualne datoteke niso izvršljive
   - moramo spremeniti dovoljenja datoteke z uporabo ukaza chmod

     ```
     $ chmod +x skripta.sh
     ```
3. Shrani skripto na lokacijo na katero je shell lahko najde
4. Za izvrševanje skripte se uporablja naslednja konstrukcija

   ```
   $ ./skripta.sh
   ```

Struktura shell skripte

```
#!/bin/bash
#Prva skripta
echo 'Hello World!'
```

- Prvi ukaz je vedno specifikacija okolice v kateri se bo izvrševala skripta

  `#!/bin/bash`

- Če ta primer ne deluje, boste morali ugotoviti, kje se nahaja vaš Bash shell. Tukaj je en način kako da ugotovite:
  $ whereis bash
- Drugi ukaz je komentar. Uporablja se znak #
- Potem sledijo ostali ukazi.

Struktura shell skripte

1. Shell skripta lahko po želji ima identifikacijsko končnico, kot je ".Sh". To pomaga le uporabniku, da bi vedel katere datoteke so katere. Procesorjev ukaz, ki je odgovoren za izvajanje datoteke uporablja izvedljivi bit, plus prvo vrstico datoteki, da bi se odločil, kako ravnati s shell skripto.
2. Običajno taka skripta se izvaja na ta način:
   $ ./scriptname.sh
   Ta poseben ukaz pove procerju da se želena skripta nahajaj v trenutnem imeniku.

Naloga: Napišite svojo prvo Shell skripto

1. Izberite urejevalnik besedila, ki ga želite uporabljati: gedit, nano, pico, emacs ali vi, ali urednika X Windows, če imate to možnost.
2. vnesite naslednje vrstice:
   #!/bin/bash
   echo "Hello, world."
3. Shranite datoteko v trenutni delovni imenik, kot "myscript.sh".
4. Premaknite se iz urejevalnika besedil v ukazni lupini.
5. V ukazni lupini, napišite: $ chmod +x myscript.sh
6. Da bi izvršili skripto, napišite: $ ./myscript.sh
   Hello, world.

Naloga

Napiši shell skripto za prikaz časa in datuma ter uporabnikov, ki so prijavljeni v sistem. Uporabljaj ukaz echo.

```
#!/bin/bash
#cas, datum in uporabniki
echo Datum in cas je naslednji:
date
echo Uporabniki ki so je prijavljeni v sistem so naslednji:
who
```

Spremenljivke

- Kot pri vseh programskih jezikih, tudi shell skriptni jezik ima spremenljivke.
- Obstaja samo eden tip spremenljivk – niz znakov
- Deklariranje spremenljivk ne obstaja.

Uporabniške spremenljivke

- Spremenljivki lahko dodelimo vrednost z uporabo znaka =

  ```
  imeSpremenljivke=vrednostSpremenljivke
  ```

- Med imenom spremenljivke in znakom =, ter med znakom = pa vrednostjo spremenljivke ne sme biti presledka
- Primeri:

  ```
  variable1=delavnica
  variable2=10
  ```

- Veljavnost uporabniških spremenljivk je dokler se skripta izvršuje.

Prikaz vrednosti spremenljivke

- Vrednost spremenljivke se pridobi z uporabo znaka $ pred imenom spremenljivke
- Primer:

```
$var1
```

- Če hočemo prikazati vrednost spremenljivke, ki ji sledi niz znakov uporabljamo velike oklepaje
- Primeri:

```
#!/bin/bash
X=ABC
echo "${X}abc"
```

```
#!/bin/bash
num=2
echo "This is ${num}nd track."
```

Struktura if–then

- Najbolj elementarna kontrolna struktura je if–then
- Sintaksa

```
if ukaz
then
    seznam ukazov
fi
```

Primer

1. Napišite:

```
if [ -e . ]
then
        echo "Yes."
else
        echo "No."
fi
```

Izvršite skripto:

```
$ ./myscript.sh
Yes.
```

Primer

Različico:

```
if test -e .
then
      echo "Yes."
else
      echo "No."
fi
```

Ne pozabite: Preberite "test" man page

```
$ man test
```

Logički testi se izvajajo na naslednem načinu:

```
$ test -e .
$ echo $?
0
$ test -e xyz $ echo $?
1
```

Zanke in ponavljanja (Loops and repetitions)

Primer:

```
for fn in *; do
      echo "$fn"
done
for fn in tom dick harry; do
      echo "$fn"
done
$ ./myscript.sh
tom
dick
harry
ls -1 | while read fn; do
      echo "$fn"
done
```

Uporaba številk v skripte

Primer

```
n=1
while [ $n -le 6 ]; do
    echo $n
    let n++
done
$ ./myscript.sh
1
2
3
4
5
6
```

Uporaba številk v skripte (2)

```
y=1
while [ $y -le 12 ]; do
    x=1
    while [ $x -le 12 ]; do
        printf "% 4d" $(( $x * $y ))
        let x++
    done
    echo ""
    let y++
done
$ ./myscript.sh
```

Secure Shell SSH

1. SSH rešuje dveh osnovnih problemov varno komunikacijo z oddaljenim gostiteljem. Prvič, authenticira, da je oddaljeni gostitelj, ki pravi, da je (s čimer bi preprečili tako imenovani "man in the middle" napadi), in drugič, da šifrira vse komunikacije med lokalnimi in oddaljenimi gostitelji.
2. SSH je sestavljen iz dveh delov. SSH strežnik teče na oddaljenem gostitelju, in posluša za prihajajoče povezave na vratih 22, medtem pa SSH odjemalec se uporablja na lokalnem sistemu, da bi komuniciral z oddaljenim strežnikom.

Povezava na superračunalniku HPCSF

1. ssh -X uporabniško_ime@prelog.fs.uni-lj.si
2. ssh -X campus31@prelog.fs.uni-lj.si
3. Poskusite: echo $SHELL, pwd, ls,...
4. Poskusite: gedit

Povezava na superračunalniku fisHPC

1. ssh -X campus31@prelog.fs.uni-lj.si

Povezava na superračunalniku fisHPC preko X2go klienta

1. http://wiki.x2go.org/doku.php/download:start
2. Download X2go za Windows OS
3. namestitev klienta....
4. Po namestitvi se za konfiguracijo "New session" izbere:
   - Session name: poljubno ime (HPC FIS ali kaj podobnega)
   - Host: 194.249.94.50
   - Login: username (user1 recimo)
5. Kot namizno grafično okolje izberite: GNOME

1. http://en.wikipedia.org/wiki/Bash_(Unix_shell)
2. http://www.arachnoid.com/linux/shell_programming.html
3. Predavanja in vaj po operaciskih sistemov, 2013-2014, FIŠ.