


# Introduction to Linux Shell

COMP201 Lab Session  
Spring 2021



**KOÇ  
UNIVERSITY**

# What is shell?

A screenshot of a Linux terminal window. The title bar at the top reads 'farzin@COMP201: /home' and includes standard window control buttons (minimize, maximize, close). Below the title bar is a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The main area of the terminal shows a green prompt 'farzin@COMP201: /home\$' followed by a white cursor. The background is dark grey.

```
farzin@COMP201: /home
File Edit View Search Terminal Help
farzin@COMP201: /home$
```

- The Linux shell is the interface between you and operating system that controls the hardware.
- The most commonly used shell is called BASH – Bourne Again Shell
- `username@hostname:curr_dir$`
  - username: farzin
  - hostname: COMP201
  - curr\_dir: /home

# Executing system programs

```
farzin@COMP201: /home
File Edit View Search Terminal Help
farzin@COMP201:/home$ date
Sun Oct 11 01:33:31 +03 2020
farzin@COMP201:/home$ echo Hello
Hello
farzin@COMP201:/home$ echo "Hello COMP201"
Hello COMP201
farzin@COMP201:/home$
```

- Execute programs
- `$date`
  - This program prints current date and time
- `$echo`
  - This program prints the input argument

# Path and \$PATH

```
farzin@COMP201: /home
File Edit View Search Terminal Help
farzin@COMP201:/home$ echo $PATH
/opt/ros/melodic/bin:/home/farzin/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
farzin@COMP201:/home$ which echo
/bin/echo
farzin@COMP201:/home$ /bin/echo Hello
Hello
farzin@COMP201:/home$ pwd
/home
farzin@COMP201:/home$
```

- \$PATH
  - A variable that contains addresses where system look for programs to execute
- \$which
  - Prints which file is being executed given an input program name
- \$pwd
  - This program prints current working directory
  - Stands for “print working directory”

# Path

```
farzin@COMP201: ~  
File Edit View Search Terminal Help  
farzin@COMP201:/home$ pwd  
/home  
farzin@COMP201:/home$ cd ~  
farzin@COMP201:~$ pwd  
/home/farzin  
farzin@COMP201:~$ cd /home  
farzin@COMP201:/home$ cd ..  
farzin@COMP201:/$ pwd  
/  
farzin@COMP201:/$ cd ./home/farzin/  
farzin@COMP201:~$ pwd  
/home/farzin  
farzin@COMP201:~$
```

- `$cd`
  - Changes the working directory
  - `..` is the parent directory
  - `.` is the current directory
  - Tilda (`~`) is the `/home/usr` directory
- Absolute vs Relative path
  - Relative: `./home/farzin`
  - Absolute: `/home/farzin`

# Listing files and directories

```
farzin@COMP201: /  
File Edit View Search Terminal Help  
farzin@COMP201:/home$ ls  
farzin  
farzin@COMP201:/home$ ls -l  
total 4  
drwxr-xr-x 44 farzin farzin 4096 Oct 11 02:02 farzin  
farzin@COMP201:/home$ cd ..  
farzin@COMP201:/ $ ls  
bin      etc          lib          media      root      srv          usr  
boot     home         lib32        mnt        run       swapfile    var  
cdrom    initrd.img   lib64        opt       /sbin     sys         vmlinuz  
dev      initrd.img.old lost+found   proc       snap      tmp         vmlinuz.old  
farzin@COMP201:/ $ ls /home  
farzin  
farzin@COMP201:/ $ ls ./home  
farzin  
farzin@COMP201:/ $
```

- \$ ls
  - Prints files and directories under current working directory
  - You can use options with commands like “-l” which shows a long list containing more details of files and folders
  - You can also pass absolute or relative path to \$ls command
  - Use --help for more info about arguments
  - Check -a and -F options

# Making directories, files, and removing them

```
fnegahbani20@WS001: ~/comp201
fnegahbani20@WS001:~/comp201$ ls
fnegahbani20@WS001:~/comp201$ mkdir my_dir
fnegahbani20@WS001:~/comp201$ ls
my_dir
fnegahbani20@WS001:~/comp201$ touch my_text.txt
fnegahbani20@WS001:~/comp201$ touch source.c
fnegahbani20@WS001:~/comp201$ ls
my_dir  my_text.txt  source.c
fnegahbani20@WS001:~/comp201$ rm source.c
fnegahbani20@WS001:~/comp201$ ls
my_dir  my_text.txt
fnegahbani20@WS001:~/comp201$ rm my_dir/
rm: cannot remove 'my_dir/': Is a directory
fnegahbani20@WS001:~/comp201$ rm -R my_dir/
fnegahbani20@WS001:~/comp201$ ls
my_text.txt
fnegahbani20@WS001:~/comp201$
```

- `$ mkdir <folder_name>`
  - Makes a new directory in the given working directory with the given “folder\_name”.
- `$ touch`
  - Creates a file with desired extension and name
- `$ rm`
  - Removes a file or folder.
  - For removing folders you need to use -R option

# File Permission in Linux

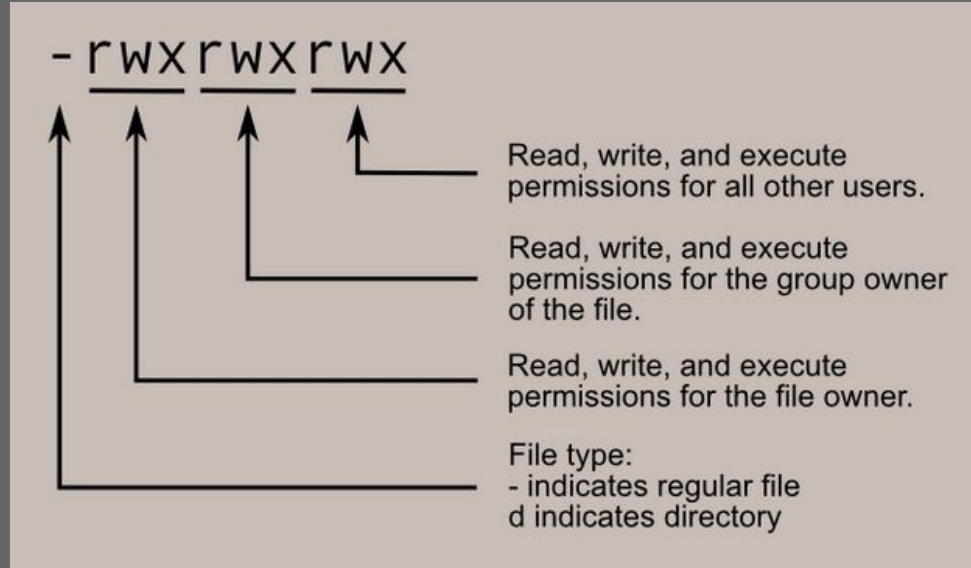


Image source: [http://linuxcommand.org/lc3\\_lts0090.php](http://linuxcommand.org/lc3_lts0090.php)



# File Permission in Linux

```
rwX rwX rwX = 111 111 111  
rw- rw- rw- = 110 110 110  
rwx --- --- = 111 000 000
```

and so on...

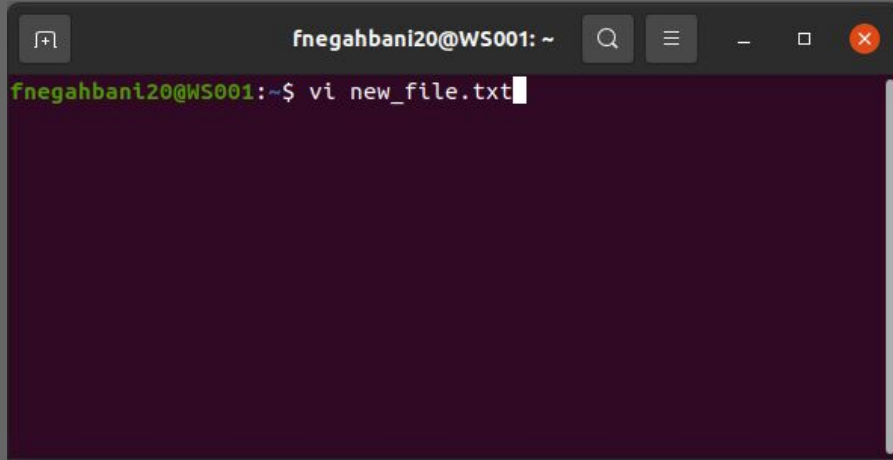
```
rwX = 111 in binary = 7  
rw- = 110 in binary = 6  
r-x = 101 in binary = 5  
r-- = 100 in binary = 4
```

Image source: [http://linuxcommand.org/lc3\\_lts0090.php](http://linuxcommand.org/lc3_lts0090.php)

Initially, test.sh cannot be executed, to grant -rwx rwx r-x permission to test.sh file:

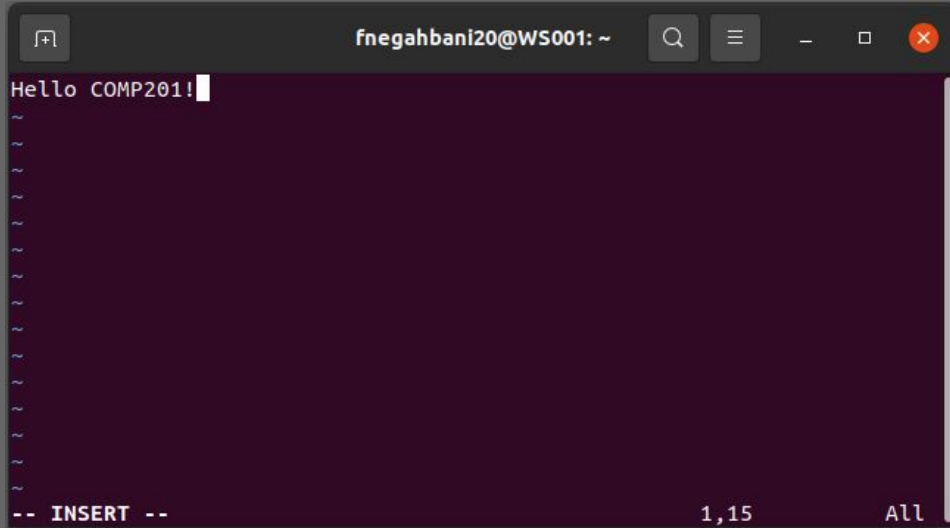
```
fnegahbani20@WS001:~$ chmod 775 test.sh
```

# What is Vi?

A terminal window with a dark background. The title bar shows 'fnegahbani20@WS001: ~' and standard window controls. The command prompt is 'fnegahbani20@WS001:~\$' and the command 'vi new\_file.txt' is being entered, with a cursor at the end of the line.


- Vi is the default text editor in the UNIX operating system.
- Using vi, we can create a new file, read, and edit an existing file.
- To open vi, type “vi” or “vi filename”. If the file “filename” doesn’t exist, it will be created when you save it.

# Operation Modes in vi or vim



- Normal mode
  - The default mode in vi.
  - In some source, like <https://www.cs.colostate.edu/helpdocs/vi.html>, it is also called command mode.
  - Every character you type is interpreted as a command.
- Insert mode
  - The one on the left picture.
  - To switch from normal mode to insert mode, type 'i' in the normal mode.
  - Every character you type is put to the file.
  - To switch back to normal mode, press <Esc>

# Operation Modes in vi or vim



The screenshot shows a terminal window with a dark background. The title bar at the top reads "fnegahbani20@WS001: ~". The terminal content shows the prompt "Hello COMP201!" followed by the user input "Let's add more" which is currently highlighted. The bottom status bar displays "-- VISUAL --" on the left, and "2", "3,15", and "All" on the right.

# Redirection

```
farzin@COMP201: ~/COMP201
File Edit View Search Terminal Help
farzin@COMP201:~/COMP201$ touch myfile.txt
farzin@COMP201:~/COMP201$ cat myfile.txt
farzin@COMP201:~/COMP201$ echo "Test1: Hello!" > myfile.txt
farzin@COMP201:~/COMP201$ cat myfile.txt
Test1: Hello!
farzin@COMP201:~/COMP201$ cat < myfile.txt
Test1: Hello!
farzin@COMP201:~/COMP201$ echo "Test2: Anybody there?" >> myfile.txt
farzin@COMP201:~/COMP201$ cat myfile.txt
Test1: Hello!
Test2: Anybody there?
farzin@COMP201:~/COMP201$ mkdir myfolder
farzin@COMP201:~/COMP201$ ls
myfile.txt  myfolder
farzin@COMP201:~/COMP201$ cat < myfile.txt > ./myfolder/myfile2.txt
farzin@COMP201:~/COMP201$ ls ./myfolder
myfile2.txt
farzin@COMP201:~/COMP201$ cat ./myfolder/myfile2.txt
Test1: Hello!
Test2: Anybody there?
farzin@COMP201:~/COMP201$
```

- `$cat`
  - Print the content of the given file
- “< file” and “> file”
  - You can wire the input and output of a program to a file
  - “>> file” appends to end of file

# Piping

```
farzin@COMP201: ~/COMP201
File Edit View Search Terminal Help
farzin@COMP201:~/COMP201$ cat myfile.txt
BaNaNA
apple
BaNaNA
orange
Apple
farzin@COMP201:~/COMP201$ cat myfile.txt | grep apple
apple
farzin@COMP201:~/COMP201$ cat myfile.txt | grep -i apple
apple
Apple
farzin@COMP201:~/COMP201$ cat myfile.txt | grep -i a
BaNaNA
apple
BaNaNA
orange
Apple
farzin@COMP201:~/COMP201$
```

- Pipe character “|”
  - Connects output of a program to input of another one
- \$grep
  - Searches for a particular information
  - By default it is case sensitive
- Try “grep --help” and find what does -i option do