

APPLIED OPERATING SYSTEM LABORATORY



FEU ALABANG



FEU DILIMAN



FEU TECH

Technology Driven by Innovation

MODULE 6

LINUX FILE OPERATIONS, REFINEMENT, AND REDIRECTORS



FEU ALABANG



FEU DILIMAN



FEU TECH

Technology Driven by Innovation

OBJECTIVES

Upon completion of this module, the student will be able to:

- Utilize different file operation, data refinement commands, and redirectors in file manipulation.

TOPIC OUTLINE

- **File Operation Commands**
 - **cp** command
 - **mv** command
 - **rm** command
 - **echo** command
 - **cmp** command
- **Data Refinement Commands**
 - **sort** command
 - **uniq** command
- **Redirectors**
 - **>**
 - **<**
 - **>>**
 - **| (Pipe)**



FILE OPERATION COMMANDS

cp command

- This allows copying file/s and directories from one location to another.

Syntax: **cp** <filename/directory> destination

Examples:

- To copy file to a directory

```
vetcha@DESKTOP-U1V5H04:~$ ls FOLDER2
file file1 file2 file3 fileA fileB fileC
vetcha@DESKTOP-U1V5H04:~$
```

```
vetcha@DESKTOP-U1V5H04: ~
vetcha@DESKTOP-U1V5H04:~$ ls
A12 FOLDER a-b c.txt empty2 file2 fileB hi.txt 'my folder' regfile1 samplevi.txt
B12 FOLDER2 a.txt empty.txt file file3 fileC his.txt names regfile3 welcome.txt
C13 a+b b.txt empty1 file1 fileA hello.txt ls numbers sample.txt
vetcha@DESKTOP-U1V5H04:~$ ls FOLDER2
vetcha@DESKTOP-U1V5H04:~$ cp f* FOLDER2
```

- To copy directory to another directory

```
vetcha@DESKTOP-U1V5H04:~$ cp -R FOLDER FOLDER2
vetcha@DESKTOP-U1V5H04:~$ ls FOLDER2
FOLDER file file1 file2 file3 fileA fileB fileC
vetcha@DESKTOP-U1V5H04:~$
```



FILE OPERATION COMMANDS

mv command

- This allows to move and rename files

Syntax: **mv** <oldname> <newname>

mv <filename/directory> destination

vetcha@DESKTOP-U1V5H04: ~/FOLDER2

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ ls
FOLDER file file1 file2 file3 fileA fileB fileC
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ mv file file4
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ ls
FOLDER file1 file2 file3 file4 fileA fileB fileC
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```

vetcha@DESKTOP-U1V5H04: ~/FOLDER2

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ ls
A FOLDER file1 file2 file3 file4 fileA fileB fileC
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ mv file4 A
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ ls A
file4
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ mv FOLDER A
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ ls A
FOLDER file4
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```



FEU ALABANG



FEU DILIMAN



FEU TECH

Technology Driven by Innovation

FILE OPERATION COMMANDS

rm command will delete a file forever.

Syntax: **rm** [-option] <filename1> <filename2>

Examples:

- To delete the file aa:

Syntax: **rm aa**

- To remove all files and all sub-directories and their contents:

Syntax: **rm -r ***

```
vetcha@DESKTOP-U1V5HO4: ~/FOLDER2
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$ ls
A aa bb cc file1 file2 file3 fileA fileB fileC
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$ rm aa
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$ ls
A bb cc file1 file2 file3 fileA fileB fileC
```

```
vetcha@DESKTOP-U1V5HO4: ~/FOLDER2
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$ ls
A cc file1 file2 file3 fileA fileB fileC
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$ ls A
cc file1 file2 file3 fileA fileB fileC
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$ rm A
rm: cannot remove 'A': Is a directory
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$ rm -r A
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$ ls
cc file1 file2 file3 fileA fileB fileC
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$
```

FILE OPERATION COMMANDS

echo command

- It displays the string or text specified after it.
- It is also used to reference and display the values of variables.
- It is commonly used in programs, or shell scripts, where user input is needed.

Syntax:

echo string

echo \$variablename

```
vetcha@DESKTOP-U1V5H04: ~/FOLDER2
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ echo Linux is Fun!
Linux is Fun!
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ x=Hello
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ echo $x
Hello
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ y="Hello World!"
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ echo $y
Hello World!
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ z=Goodbye
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ echo $z
Goodbye
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ echo $y $z
Hello World! Goodbye
```



FEU ALABANG



FEU DILIMAN



FEU TECH

Technology Driven by Innovation

FILE OPERATION COMMANDS

cmp command

- It checks two files to see if they differ.
- It does a byte-by-byte comparison of file1 and file2.
- If the files differ, **cmp** outputs the **location** at which the **first difference** occurs.

Syntax: **cmp** [options] <filename1> <filename2>

```
vetcha@DESKTOP-U1V5HO4: ~/FOLDER2  
  
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$ cat a b  
The quick brown fox jumped over the lazy dog's back.  
The quick brown fox jumped over the lasy dog's back.  
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$ cmp a b  
a b differ: byte 39, line 1  
vetcha@DESKTOP-U1V5HO4:~/FOLDER2$
```



FEU ALABANG



FEU DILIMAN



FEU TECH

Technology Driven by Innovation

DATA REFINEMENT COMMANDS

sort command

- Sorts and/or merges one or more text files in sequence.

Syntax: **sort** [option] <filename>

sort [option] <filename1> <filename2>

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat days
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ sort days
Friday
Monday
Saturday
Sunday
Thursday
Tuesday
Wednesday
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ sort -r days
Wednesday
Tuesday
Thursday
Sunday
Saturday
Monday
Friday
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ sort -R days
Saturday
Tuesday
Thursday
Monday
Sunday
Wednesday
Friday
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ sort -R days
Wednesday
Monday
Saturday
Sunday
Thursday
Friday
Tuesday
```



DATA REFINEMENT COMMANDS

uniq command

- It reports or **filters** out the **repeated lines** in a file.
- It is the tool that helps to detect the **adjacent duplicate lines** and also **deletes** the duplicate lines.
- Ensures that **no two lines** that it displays are **the same**.

Syntax: **uniq** [option] <filename>

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat>fruits
apple
apple
banana
banana
lemon
chico
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ uniq fruits
apple
banana
lemon
chico
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ uniq -c fruits
  2 apple
  2 banana
  1 lemon
  1 chico
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ uniq -d fruits
apple
banana
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ uniq -D fruits
apple
apple
banana
banana
```



DATA REFINEMENT COMMANDS

grep command

- Used for pattern searching.
- Users can use this command to search a set of files for one or more phrases or patterns.
- If the pattern exists, then **grep** will print all the lines that contain the said pattern.

```
vetcha@DESKTOP-U1V5H04:~$ cat > names
Joed Goh
Luvett Goh
Sam Goh
Jewel Goh
vetcha@DESKTOP-U1V5H04:~$ grep Goh names
Joed Goh
Luvett Goh
Sam Goh
Jewel Goh
vetcha@DESKTOP-U1V5H04:~$ grep goh names
vetcha@DESKTOP-U1V5H04:~$
```

Syntax: **grep pattern <filename>**

where:

pattern is the phrase or pattern the user wants to find.

filename is the name of the target file.



FEU ALABANG



FEU DILIMAN



FEU TECH

Technology Driven by Innovation

REDIRECTORS

- **Redirection** is a feature in Linux such that when executing a command, you can **change the standard input/output devices**.
- The basic workflow of any Linux command is that it takes an input and give an output.
- The standard input (stdin) device is the keyboard.
- The standard output (stdout) device is the screen.
- There are three main redirection symbols:
 - **>** - Output redirection
 - **<** - Input redirection
 - **>>** - Append
 - **|** - Pipe



FEU ALABANG



FEU DILIMAN



FEU TECH

Technology Driven by Innovation

REDIRECTORS

> (Output Redirection)

- Using this redirector symbol, the output from a command normally intended for standard output can be easily diverted to a file instead.

Note: If file already exist, it will be overwritten, else new file is created.

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat days
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat days > week1
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat week1
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
```

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat fruits
apple
apple
banana
banana
lemon
chico
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ ls > fruits
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat fruits
a
b
b.txt
cc
days
file1
file2
file3
fileA
fileB
fileC
fruits
week1
```



REDIRECTORS

< (Input Redirection)

- The less-than character < is used to redirect the input of a command.

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat days
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat < days
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat days
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ wc -l days
7 days
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ wc -l < days
7
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```



FEU ALABANG



FEU DILIMAN



FEU TECH

Technology Driven by Innovation

REDIRECTORS

>> (Append)

- The >> operator redirects output to a file, if the file doesn't exist, it is created but if it exists, the output will be appended at the end of the file.

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat numbers
one
two
three
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ echo four >> numbers
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat numbers
one
two
three
four
```

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ date >> numbers
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat numbers
one
two
three
four
Tue Jul 28 23:06:21 PST 2020
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ echo hello >> hello.txt
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat hello.txt
hello
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ echo goodbye >> hello.txt
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat hello.txt
hello
goodbye
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```



REDIRECTORS

| (Pipe Symbol)

- A pipe is the same as redirecting standard output.
- It is nothing but a **temporary storage** place where the output of one command is stored and then **passed as the input for second command**.
- Pipes are used to run more than two commands (multiple commands) from same command line.

Syntax: **command1 | command2**

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat days
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ sort days | cat > sorted_days
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat sorted_days
Friday
Monday
Saturday
Sunday
Thursday
Tuesday
Wednesday
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat hello.txt
hello
goodbye
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat hello.txt | cat > h.txt
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat h.txt
hello
goodbye
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```

```
vetcha@DESKTOP-U1V5H04:~/FOLDER2$ cat days | head -2 | grep day
Monday
Tuesday
vetcha@DESKTOP-U1V5H04:~/FOLDER2$
```



REFERENCES

- Sobell, M., et al. (2017). A Practical Guide to Linux Commands, Editors, and Shell Programming, 4th Ed. Addison-Wesley Professional
- Cobbaut, P. (2016). Mastering Linux- Networking
- Blum, R., (2015). Linux Command Line and Shell Scripting Bible



FEU ALABANG



FEU DILIMAN



FEU TECH

Technology Driven by Innovation