

East West University
Department of Computing Science and Engineering
CSE-325: Operating System, Lab-1

Objective: Working with file and directory commands at Linux command line.

Absolute Path: An absolute path is defined as the specifying the location of a file or directory from the root directory (/). In other words, we can say absolute path is a complete path from start of actual filesystem from / directory.

Example: */home/Aftabnagar/EWU/CSE325/Lab/Lab 1/CSE325_Lab_1.pdf* and *~/EWU/CSE325/Lab/Lab 1/CSE325_Lab_1.pdf* are absolute paths of the file CSE325_Lab_1.pdf.

Relative Path: Relative path is defined as path with respect to the present working directory(pwd).

Example: In the previous example of the file CSE325_Lab_1.pdf, if current directory is EWU, the relative path of the pdf file is *CSE325/Lab/Lab 1/CSE325_Lab_1.pdf* or *./CSE325/Lab/Lab 1/CSE325_Lab_1.pdf*.

Commands

pwd (Prints the current working directory)

Syntax: pwd

mkdir (Creates new directories)

Syntax: mkdir *dir_path*

Example:

mkdir ABC

mkdir ~/Pictures/Food

mkdir ABC/DEF GHI ~/Videos/PQR

rmdir(Deletes empty directories)

Syntax: rmdir *dir_path*

Example:

```
rmdir XYZ  
rmdir /home/kowshika/Videos/Picnic  
rmdir ~/Videos/PQR GHI
```

cd (Changes current directory)

Syntax: `cd dir_path`

Example:

```
cd ABC  
cd ./ABC/DEF  
cd ~/Pictures/Food
```

ls (Lists file and directory information)

Syntax: `ls [options] [path]`

Example:

```
ls  
ls -l  
ls -R  
ls ./File.txt  
ls -lR ~/Pictures/Picnic  
ls -lR ~/Pictures/Picnic >> list.txt
```

cat

Syntax:

`cat >> file_path`

[Will create & write contents to a file. After the command, type file contents following by Ctrl+D]

`cat file_path`

[Will display the content of the file]

`cat file1_path file2_path ... fileN_path >> file_name`

[Will merge the contents from file1, file2, ..., fileN into a]

Example:

```
cat >> file1  
cat >> ~/ABC/file2.txt  
cat file1  
cat file1 ~/ABC/file2.txt  
cat file1 ~/ABC/file2.txt >> file3
```

cp (Copies file)

Syntax: `cp source_path destination_path`

Example: `cp file3 ~/Documents/file4.txt`

mv (Renames/moves file)

Syntax: `mv source path destination path`

Example: `mv ~/Documents/file4.txt ./ABC/file5`

sort (Sorts contents of file according to ASCII number)

Syntax: `sort file_path`

Example:

`sort ~/Documents/file4.txt`

`sort ~/Documents/file4.txt > ./ABC/sort.txt`

`sort ~/Documents/file4.txt >> ./ABC/sort.txt`

chmod (Changes permission of file or directory)

All file system objects on Unix-like systems have three main types of permissions: read, write, and execute access. Permissions are bestowed upon three possible classes: the user, the user group, and all system users. After print `ls -l` we can see in the first column of the output, there are 10 characters that represent the permission bits. The first character represents the type of file. The remaining nine bits in groups of three represent the permissions for the user, group, and all system users.

Syntax: `chmod permission_bits file_name`

Example: `chmod u=rwx,g=rw,o=r ./ABC/sort.txt`

umask (Sets default permission)

Syntax: `umask`

Example:

`umask u+w`

`umask u-x,g=r,o+w`

permission symbol is any combination of r (read), w (write), or x (execute).

user class symbols are u(user), g(group), o(others), a(all).

permissions operators are +, -, =.

+ (allow the specified file permission to be enabled for the specified user classes)

- (prohibit the specified file permission from being enabled for the specified user classes)

= (allow the specified file permission to be enabled for the specified user classes)

Any command output can be appended to a file using >> operator

Any command output can be written to a file using > operator

Practice

Open a terminal in Home directory.

1. Create a directory named *Homework*.
2. Position your session in the *Homework* directory.
3. Create two more sub-directories named *Homework-1* & *Homework-2*.
4. Position your session in the *Homework-1* directory.
5. Create 2 files named *1.txt* and *2.txt* and write two lines regarding your interest in each text file.
6. Merge these 2 files into one file named *merge.txt*.
7. Change permission for *merge.txt* so that user has all 3 permissions and user group has only read and write permission and others have only read permission.
8. Do a directory listing to see the contents of *Homework-1* folder.
9. Move to *Homework-2* directory.
10. Change default permission such that execute permission is added to user, write permission is denied from group and others have only read permission.
11. Create a file named *3.txt* to and write 5 random lines in the file.
12. Sort the file and write the sorted output in a file named *sort.txt* inside *Homework-1* directory.
13. Rename *merge.txt* to *combine.txt*.
14. Merge *sort.txt* and *combine.txt* in a new file named *final.txt* inside *Homework-2* directory.
15. Move *final.txt* from *Homework-2* to *Homework* directory.
16. Recursively list the contents of *Homework* directory.
17. Change your session to *Homework* directory.
18. Display the contents of *final.txt*.