Study Guide

List of commands

date command

Description:

Displays the current time and date

Syntax:

```
date
```

Example:

- Displays date and time
    - `date`

uname command

Description:

Displays information about your system

Syntax:

```
uname
```

Example:

- Displays information in system
    - `uname`

du command

Description:

Displays the disk usage

Syntax:

```
du
```

Example:

- Displays disk usage
    - `du`

free command

Description:

Displays the amount of free memory

Syntax:

```
free
```

Example:

- Displays free memory
    - `free`

echo command

Description:

Displays a line of text or string as a standard output

Syntax:

```
echo + "option"
```

Example:

- Displays hello world!
    - `echo "hello world!"`
- Displays i like cats...
    - `echo "i like cats..."`
- Displays ice cream is the bomb
    - `echo "ice cream is the bomb"`

apt command

Description:

It is a set of tools that help manage all packages in the system

Syntax:

```
sudo apt install + option -y
```

Example:

- Command for updating Ubuntu
    - `sudo apt update; sudo apt upgrade -y`
- Command for installing software
    - `sudo apt install firefox -y`
- Command for removing software
    - `sudo apt remove firefox -y`

pwd command

Description:

Used to display the current working directory

Syntax:

```
pwd
```

Example:

- Displays current working directory
    - `pwd`

cd command

Description:

Used to change the current working directory

Syntax:

```
cd + destination
```

Example:

- Changes directory to Documents
    - `cd ~/Documents`
- Changes directory to Word inside Documents
    - `cd ~/Documents/Word`
- Changes directory back to home
    - `cd`

ls command

Description:

Used to display a list of all the files inside the current working directory

Syntax:

```
ls
```

Example:

- Displays a list of all files
  - `ls`
- Displays a long list of all file
  - `ls -l`
- Displays a long list without the user or group
  - `ls -lGg`

tree command

Description:

used to display a list of all files inside the current directory in a tree form

Syntax:

```
tree
```

Example:

- Displays a list of files in tree form
  - `tree`

man command

Description:

Displays the user manual of any command run in the terminal

Syntax:

```
man + command name
```

Example:

- Displays the manual for ls
  - `man ls`
- Displays the manual for all commands
  - `man`

mkdir command

Description:

Used for creating a single directory ot multiple directories

Syntax:

```
mkdir + name of directory
```

Example:

- Creates a directory in the present working directory
  - `mkdir wallpapers`
- Creates a directory and uses a space to name it
  - `mkdir lollipops candy`
- Creates multiple directories
  - `mkdir wallpapers/guitars wallpapers/iceCream`

touch command

Description:

Used to create files

Syntax:

```
touch + name of file
```

Example:

- Creates a file
  - `touch CocaCola`
- Creates multiple files
  - `touch CocaCola.docx Sprite.txt`
- Creates a file inside a directory
  - `touch Downloads/Stars.txt`

rm command

Description:

Used to remove files

Syntax:

```
rm + name of file
```

Example:

- Removes file
  - `rm CocaCola`
- Removes multiple files
  - `rm CocaCola.docx Sprite.txt`
- Removes an empty directory
  - `rm Downloads/Stars.txt`

cp command

Description:

Used to copy files/directories from a source to a destination

Syntax:

```
cp + files to copy + destination
```

Example:

- Copies a file
  - `cp Downloads/swim.png Pictures/`
- Copies multiple files
  - `sudo cp -r swim.png car.jpeg Downloads/ Pictures/`
- Copies file of a directory to another directory
  - `cp Downloads/wallpapers/swim.png/* ~/Pictures/`

mv command

Description:

Moves and renames directories

Syntax:

```
mv + source + destination mv + file/directory to rename + new name
```

Example:

- Moves a file from a directory
  - `mv Downloads/hw.pdf Documents/`
- Moves multiple files to a different directory
  - `mv music/ games/ tricks/ ~/Media/`
- Renames a file
  - `mv hw.pdf cis106hw.pdf`

stat command

Description:

A data structure contains all the information about a file minus the file name and what it contains

Syntax:

```
stat + file name
```

Example:

- Displays data of file
    - `stat word.txt`
- Displays where the file resides, no information about the files
- `stat -f word.txt`

Wildcard (*)

Description:

Matches anything and nothing and matches any number of characters

Syntax:

```
ls *end of any file
```

Example:

- Lists all the files ending in .txt
    - `ls *.txt`
- Lists all files ending in multiple extensions
    - `ls *.txt *.docx`

Wildcard (?)

Description:

Matches precisely one character

Syntax:

```
ls .??*
```

Example:

- Lists all files starting with two characters
    - `ls .??*`
- Lists hidden files starting with two characters in current working directory
    - `ls ./.??*`

Wildcard []

Description:

Matches a single character in a range

Syntax:

```
ls *? *[a-z] *
```

Example:

- Matches all files with a vowel after the letter F
  - `ls F[aeiou]*`
- Matches all files starting with any number between 1 and 10
  - `ls [1-10]*`
- Matches all files with a lowercase letter after the 3rd character
  - `ls *??[a-z] *`

Brace Expansion {}

Description:

NOT a wildcard! This allows you to generate arbitrary strings to use with commands

Syntax:

```
touch + file{A..Z}.txt
```

Example:

- Creates a whole directory structure in a single command
  - `mkdir -p music/{ccm,love}/{mp3files,videos}/new{1..3}`
- Creates a N number of files
  - `touch file{{a..z},{0..10}}.js`
- Removes multiple files in a single directory
  - `rm -r {dir1,dir2,file.txt,file.py}`

cat command

Description:

Displays the content of a file, short for concatenate

Syntax:

```
cat + option + file(s) to display
```

Example:

- Displays the content of a file in pwd
  - `cat todo.md`
- Displays content of a file with line numbers
  - `cat -n ~/Documents/todo.md`
- Displays content of a file suppressing repeating empty lines to a single empty line
  - `cat -s ~/Documents/todo.md`

head command

Description:

Displays the first N number of lines of a given file

Syntax:

```
head + option + file
```

Example:

- Displays the first 5 lines of a file
  - `head -5 ~/Documents/Books/dracula.txt`
- Displays the first 10 lines of a file
  - `head ~/Documents/Books/dracula.txt`

tail command

Description:

Displays the last N number of lines of a given file

Syntax:

```
tail + option + file
```

Example:

- Displays the last 5 lines of a file
  - `head -5 ~/Documents/Books/dracula.tx`
- Displays the last 10 lines of a file
  - `head ~/Documents/Books/dracula.txt`

cut command

Description:

Used to extract a specific section of each line of a file and display it on the screen

Syntax:

```
cut + option + file(s)
```

Example:

- Displays a list of all the users in the system
  - `cut -d ";" -f1 /etc/passwd`
- Uses a delimiter but changing the delimiter in the output
  - `cut -d ";" -f1,7 --output-delimiter=' => ' /etc/passwd`
- Excludes a given field
  - `cut -d ";" --complement -s -f3 users.txt`

tr command

Description:

Used to translate or delete characters from a standard output

Syntax:

```
standard output | tr + option + set + set
```

Example:

- Translates one character int another
  - `cat file.txt | tr "." ","`
- Translates a white space into tabs
  - `cat file.txt | tr "[:space:]" '/t'`
- Translates tabs into a space
  - `cat file.txt | tr -s "[:space:]" ' '`

paste command

Description:

Used to join files horizontally in columns

Syntax:

```
paste + option + file(s)
```

Example:

- Merge two files
  - `paste users.lst ip_address.lst`

- Merge tow files using a different delimiter
    - `paste -d ";" users.lst ip_address.lst`

wc command

Description:

Used to print the number of lines, characters, and bytes in a file

Syntax:

```
wc + option + file(s)
```

Example:

- Displays the number of characters in a file
    - `wc -n users.txt`
- Displays the number of lines in a file
    - `wc -l users.txt`
- Displays the number of words in a file
    - `wc -w users.txt`

grep command

Description:

Used to search a text in a given file

Syntax:

```
grep + option + search criteria + file(s)
```

Example:

- Search any line that contains the word "dracula" in the given file
    - `grep "dracula" ~/Documents/dracula.txt`
- Search any line that contains the word "dracula" regardless of the case
    - `grep -i "dracula" ~/Documents/Books/dracula.txt`
- Search and display the total number of times a given word appears
    - `grep -wc "bin/bash" /etc/passwd`

output redirection (>)

Description:

Used to save or redirect the output of a command in another file inside the system

Syntax:

```
command output + > + file
```

Example:

- Save the output of a command to a file
    - `ls -lA ~ > all-files-in-home.txt`
- Save the error generated by a command to a file
    - `ls -lA Downloads/ 2> error-of-ls`
- Save the error and success to the same file
    - `ls -lA Downloads/ Pictures > success.txt 2> error.txt`

Saving the output of a command

Description:

Saves the output of a command in another file inside the system

Syntax:

```
command output + > +file
```

Example:

- Save the output of a command to a file
    - `ls -lA ~ > all-files-in-home.txt`

vim command

Description:

A text editor

Syntax:

```
vim + option
```

Example:

- Open a file
    - `vim stars.txt`
- Close a file
    - `vim :q`
- Edit a file
    - `vim :e stars.txt`

tar command

Description:

Creates archives by combining files and directories into a single file

Syntax:

```
tar + options + archive name + files to add to archive tar + options + files to extract
```

Example:

- Creates an archive
  - `tar -cf ex.tar Exam file1 file2`
- Extracts archive
  - `tar -xf ex.tar`
- Lists the contents of an archive
  - `tar -tf ex.tar`

gz, bzip2, or xz

Description:

Used for compression **bzip2** - offers better compression ratios in comparison to gzip **xz** - produces better compression ratio that both gzip and bzip2

Syntax:

```
gzip file.txt
```

Example:

- Compress a single file'
  - `gzip file.txt`
- Compress and keep the file
  - `bzip2 -k file.txt`
- Decompress a file
  - `xz -d file.txt.xz`

chmod

Description:

Used to change the access permissions of a file system, read, write, and execute

Syntax:

```
chmod + rwx + file
```

Example:

- Allow permission for everyone
    - `chmod rwx Documents/`
- Allow permission only to read
    - `chmod r- Documents/`
- Allow permission only to write
    - `chmod w- Documents/`