SIRE511 : LINUX AND BIOINFORMATICS DATA SKILLS

Fundamental Linux PART II :

2nd Week, 09/09/2024

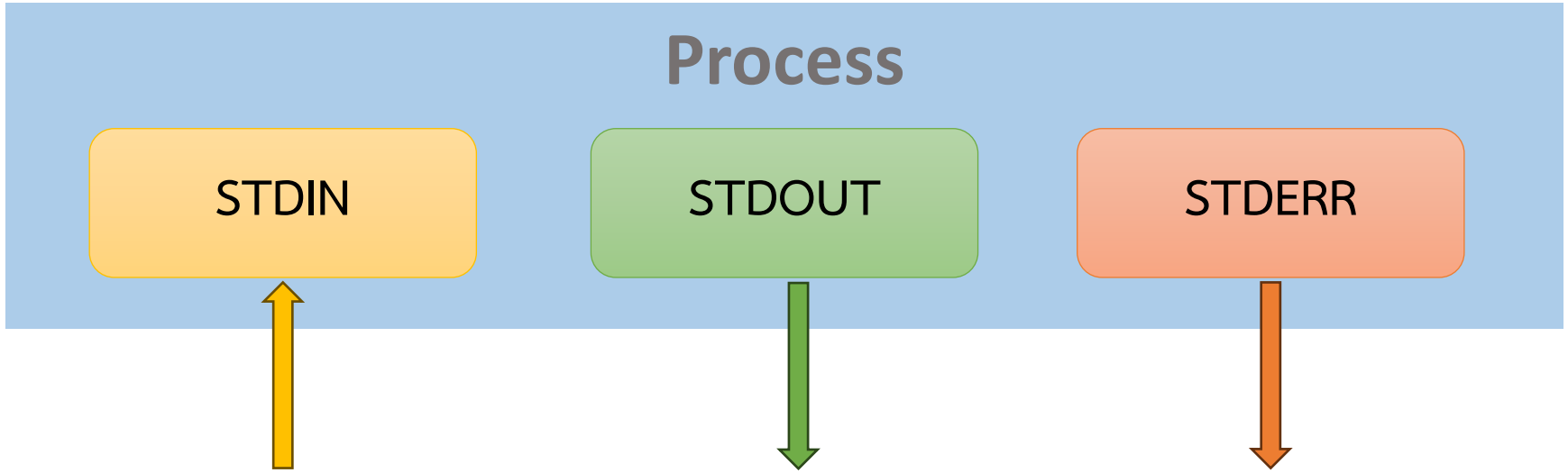
Kwanrutai Mairiang, Ph.D

Outline:

- Data streams and piping in Linux
- Linux file system
- Files and directories management
- Editing files using text editor
- Soft and Hard links

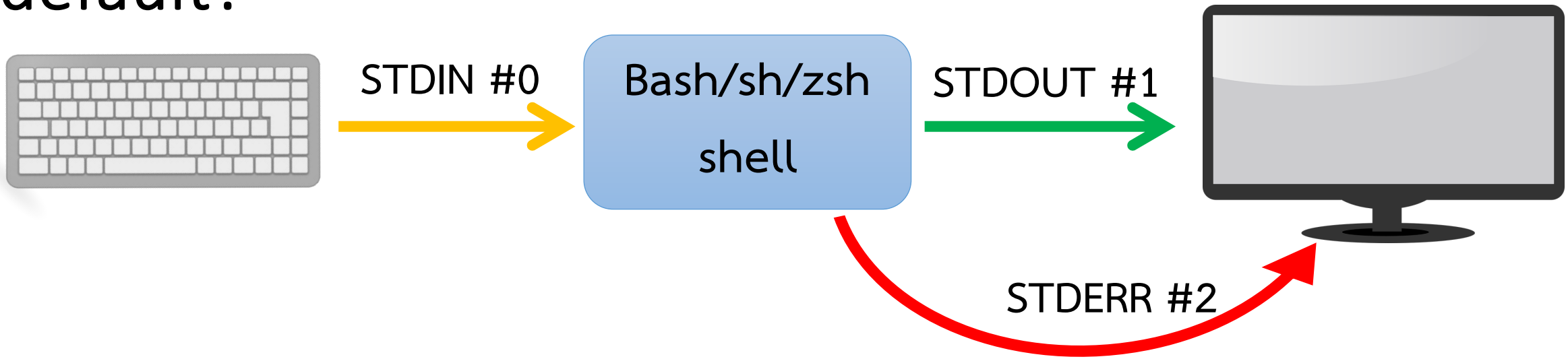
Data streams and piping in Linux

Data streams of the process



- When the process is running, three different data streams are present:
  - STDIN = Standard In, inbound data stream that enters the process.
  - STDOUT = Standard Out, outbound data stream.
  - STDERR = Standard Error, outbound data stream.

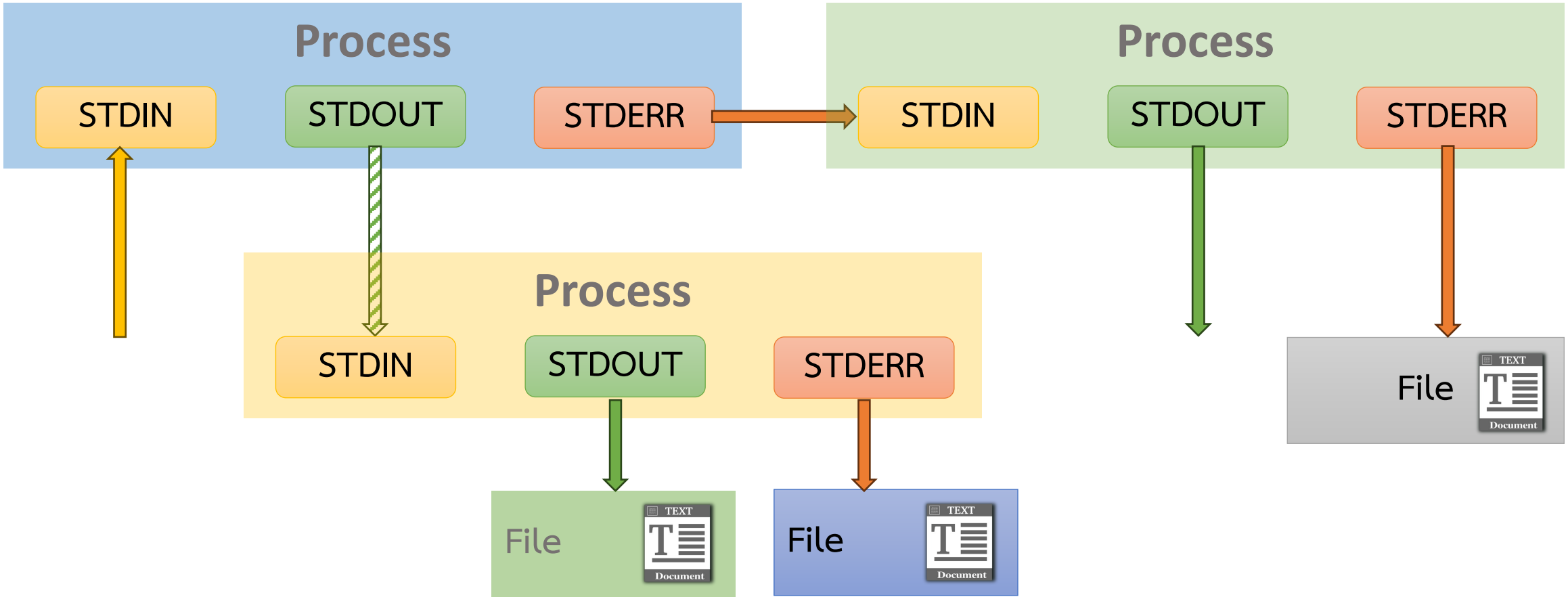
Where do process data streams send data by default?



- STDIN serves as the input data source. By default, it captures any text entered from the keyboard.
- STDOUT represents the command's output. By default, this output is shown on the screen.
- STDERR contains error messages generated by commands. By default, these error messages are also shown on the screen.

https://linuxhandbook.com/redirection-linux/

How to send data to STDIN and redirect STDOUT and STDERR to file



Redirecting STDOUT and STDERR to file

- Redirection Operators of STDIN, STDOUT, and STDERR
  - The “<” or “0<” is used for the STDIN stream.
  - The “>” or “1>” is used for the STDOUT stream.
  - The “2>” is used for the STDERR stream.
- Redirect the data stream by appending the data to the same file using “>>”.
  - The ">>" is used to append STDOUT to the same file.
  - he "2>>" is used to append STDERR to the same file.

https://linuxhint.com/bash_stdin_stderr_stdout/

Practice: Redirect process data stream

- Run the following “cat” command to create a text file named testdata.txt

```
$ cat > testdata.txt
$ cat >> testdata.txt
$ cat < testdata.txt
```

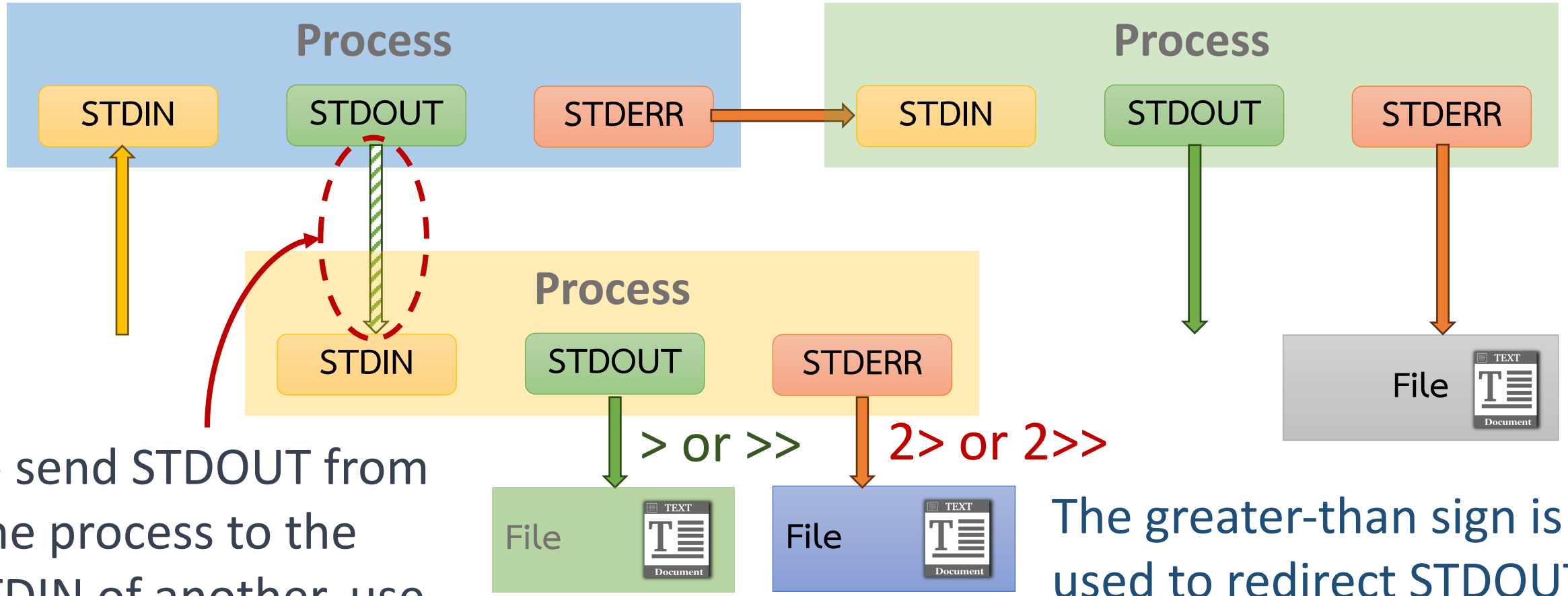
- STDOUT
  - Run the following command to write the **output** of the “ls -l” command into a text file

```
$ ls -l > list.txt
```

- STDERR
  - Run the following command to write the **error** of the “ls” command into a text file

```
$ ls -l list.tx
$ ls -l list.tx 2> ls_error.txt
```

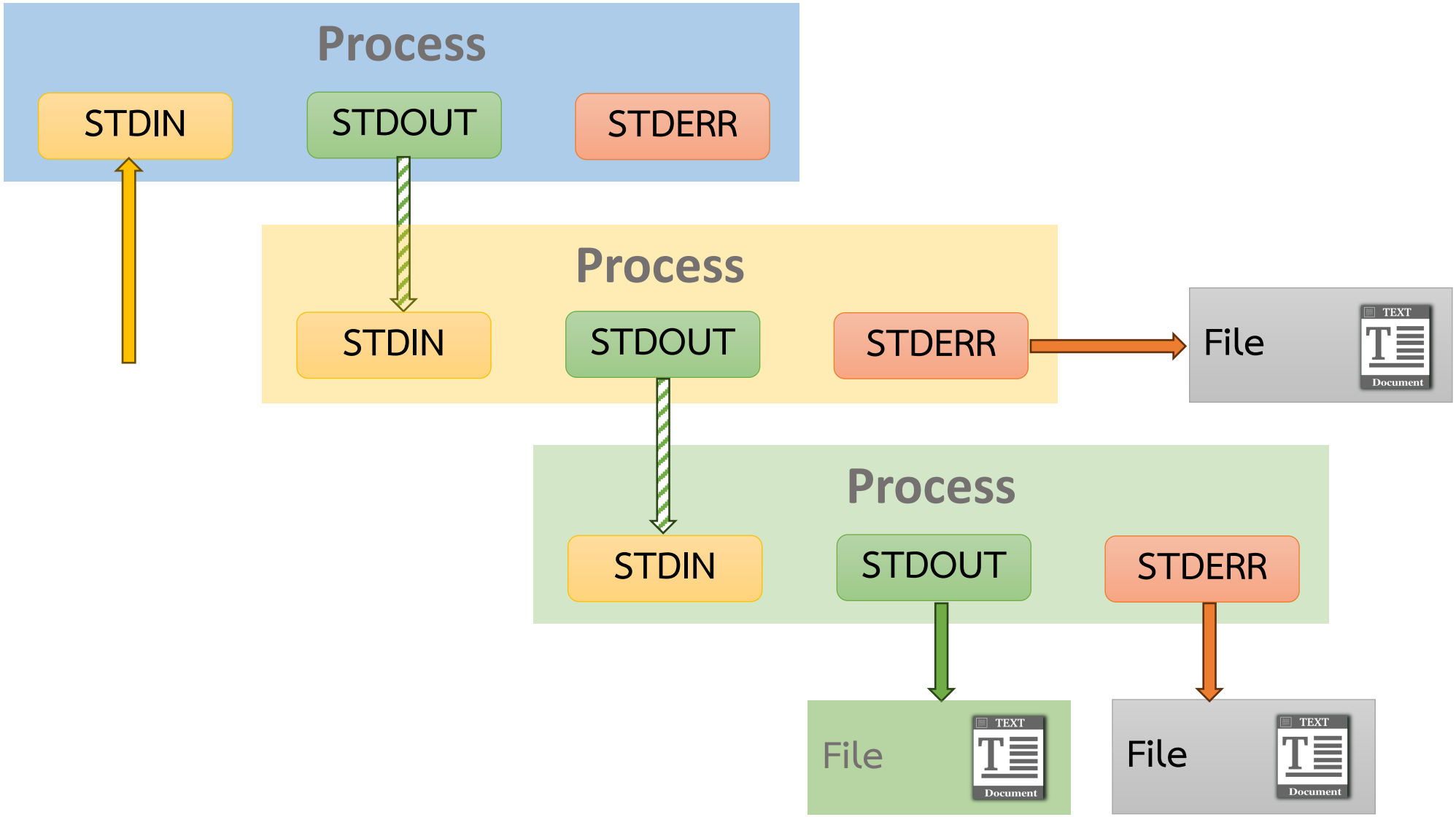
Piping: to send the output from one process to another



To send STDOUT from one process to the STDIN of another, use the pipe symbol "|".

The greater-than sign is used to redirect STDOUT and STDERR to a file.

Piping


Process
STDIN
STDOUT
STDERR
Process
STDIN
STDOUT
STDERR
File
TEXT
Document
Process
STDIN
STDOUT
STDERR
File
TEXT
Document
File
TEXT
Document

Running multiple Linux commands

- Running command consecutively
  - Running the next command after the previous one has finished.

```
$ command1 ; command2 ; command3
```

- Executing commands in parallel
  - Running commands concurrently or in parallel. Commands 1, 2, and 3 are executed at the same time.

```
$ command1 & command2 & command3
```

- Using the logical operators
  - The logical operators && and || are used to run commands based on the success or failure of the previous process.

```
$ command1 && command2    ## Command 2 will run if command 1 is successful.
$ command1 || command2    ## Command2 will run if command1 fails.
```

- Grouping commands
  - Parentheses can be used for grouping commands when you want to execute them in a specific order.

```
$ (command1 & command2) & command3
```

Commands 1 and 2 will run simultaneously, but command 3 will run after commands 1 and 2 have finished.

https://www.linuxjournal.com/content/mastering-terminal-command-execution-running-multiple-linux-commands-simultaneously

Practice: Piping

- Try running the following command.
  - Using “>” symbol
    - `$ ls > wc -l`
  - Using “|” symbol
    - `$ ls | wc -l`
    - `$ ls -la | wc -l`
    - `$ ls -la | cut -d " " -f 1`
    - `$ ls -la | cut -d " " -f 3 | uniq -c`

Summary for the data stream and piping

- Every process has three distinct data streams: STDIN, STDOUT, and STDERR.
- STDIN is the incoming data stream, while STDOUT and STDERR are the two distinct outgoing data streams from every process.
- By default, the outgoing data streams of STDOUT and STDERR are displayed on the screen, but you can redirect them to a file.
- Different processes can be concatenated by piping the STDOUT from a previous process to the STDIN of the next process.

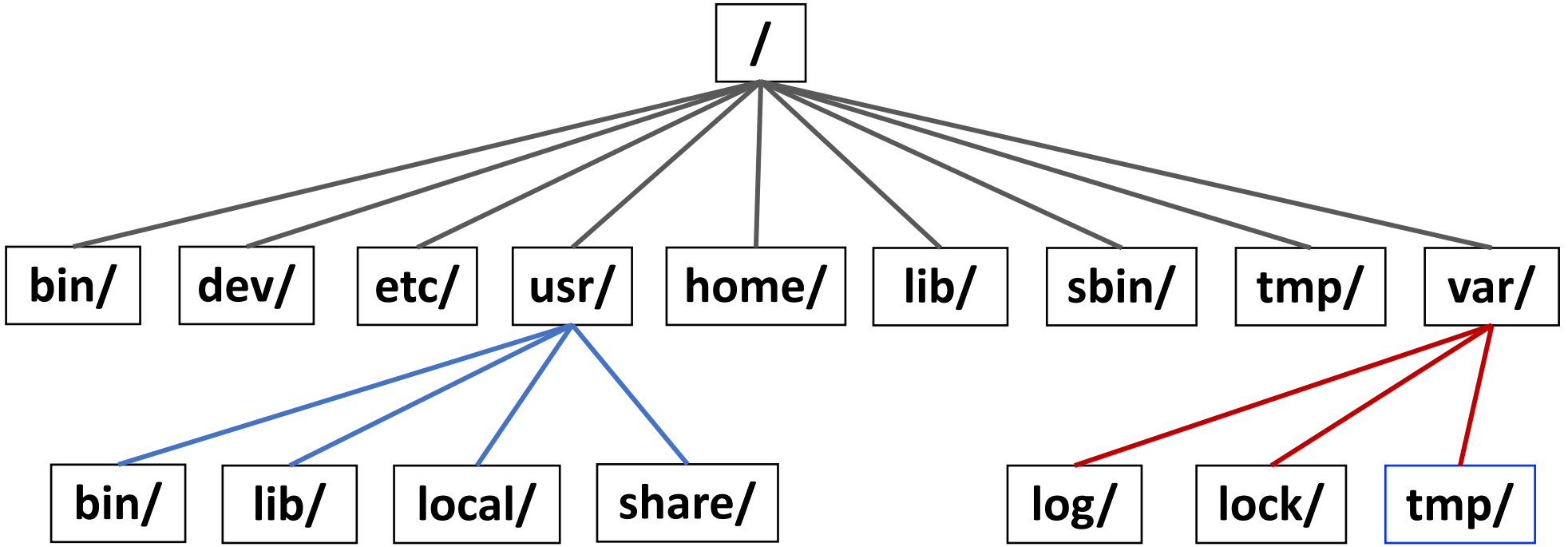
Files and directories management in Linux

What will we learn from this topic?

- Linux file system
  - Linux File Hierarchy Structure
  - How to move through different directories in Linux?
- How to manage file and directories?
  - Create new directories
  - Create new files
  - Remove files
  - Remove directories
  - Rename files and directories
  - Moving file and directories

Linux file system

Linux file system structure and navigation



- The structure looks similar to an upside-down tree.
- All the files and directories in Linux are located under root (/).
- All directories have a parent except for the root directory.
- Pathname was used to refer to files and directories.

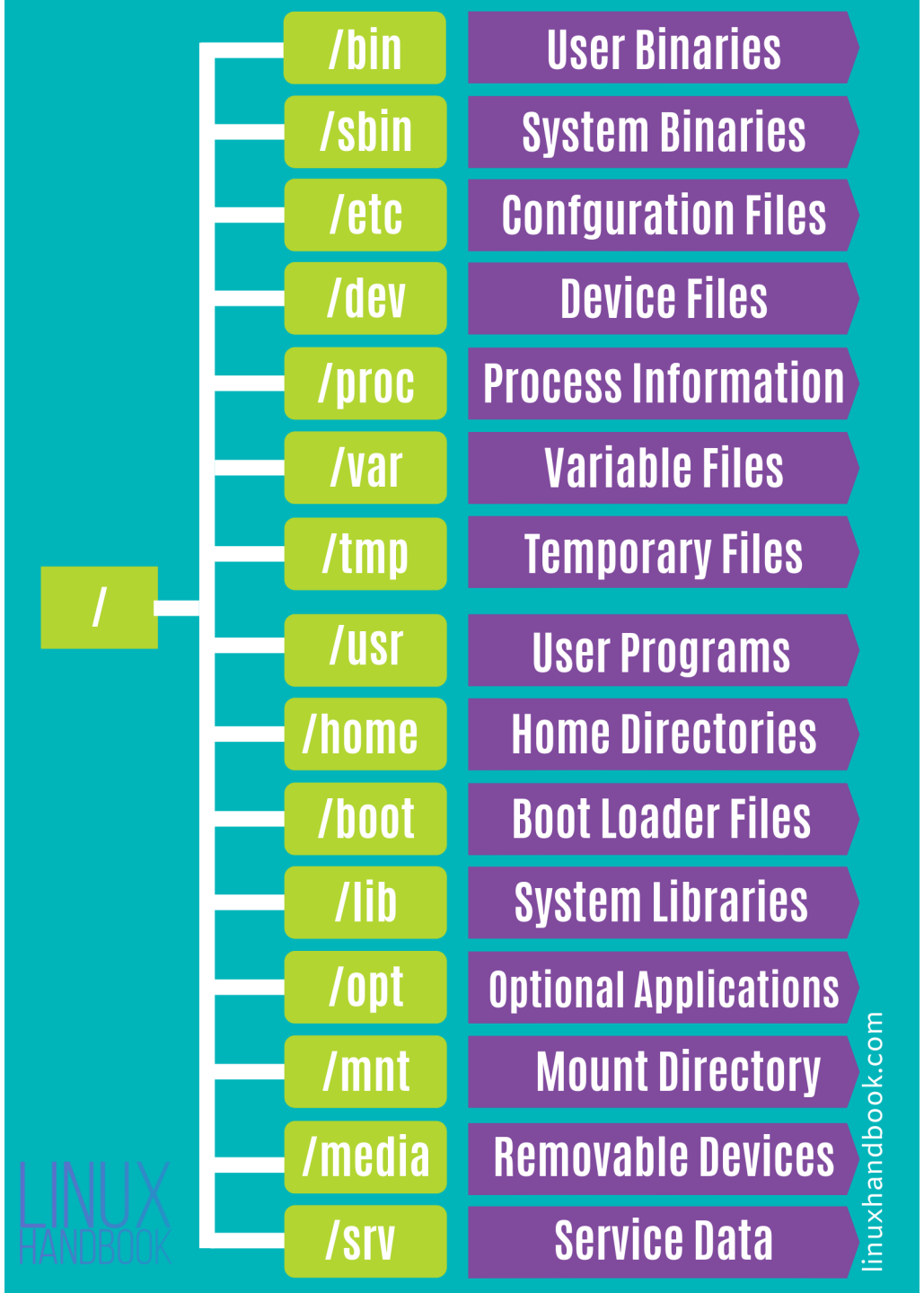
What is the full path of folder “tmp” ?
Root --> var --> tmp
Full path = /var/tmp

https://linuxhandbook.com/linux-directory-structure/

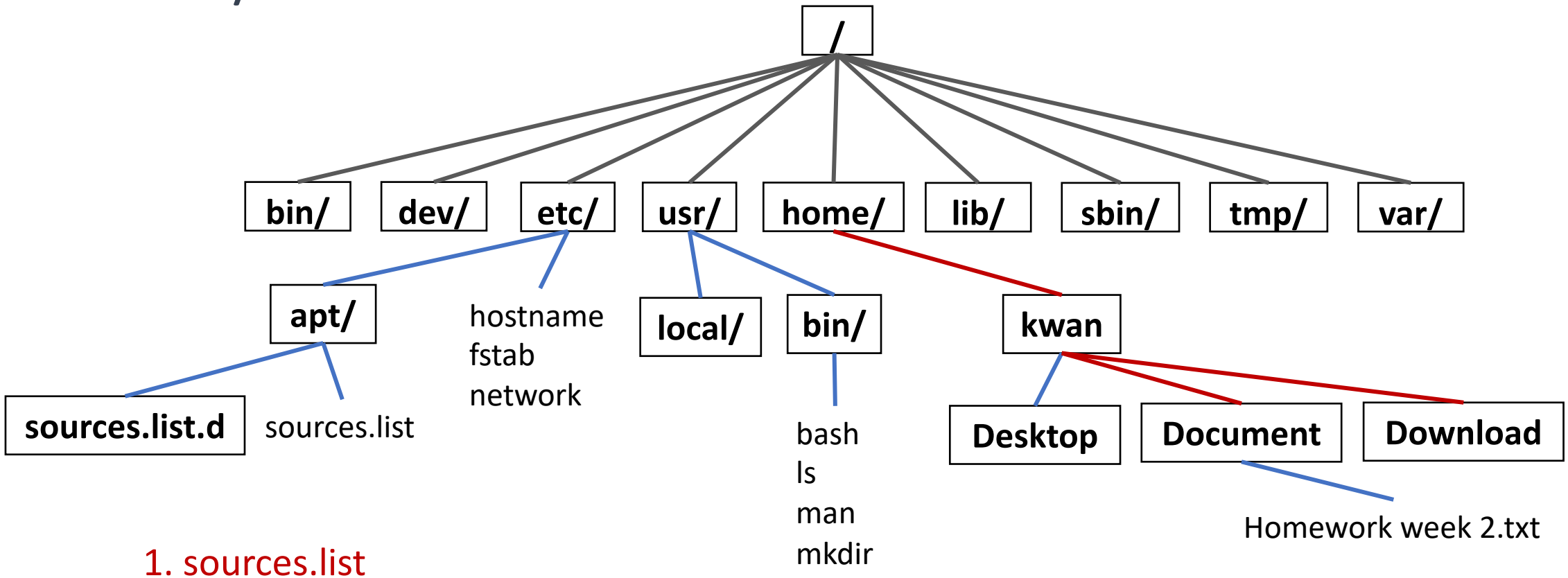
Quick reference to the directory structure in Linux systems.

Directory	Description
/bin	It contains all essential user binary files needed for the boot process and to run the system in single-user mode, including essential commands such as cd, ls, cp, etc.
/sbin	It contains applications that only the super user will need. This directory contains tools that can install, delete and format. The application here need to be run with “sudo” command.
/etc	It contains all the configuration files required by the system and other applications.
/dev	It contains all hardware devices file on the system, representing hardware components or driver.
/tmp	A repository for temporary files created by applications and users. These files are usually deleted upon reboot or after a set period.
/usr/bin	Contains commands available to normal users
/usr/share	Contains shared directories for man files, info files, etc.
/usr/sbin	Commands used by the super user for system administrative functions
/home	User home directories
/boot	Holds files used during the boot process along with the Linux kernel itself
/lib	Contains essential libraries needed for the binaries in “/bin” and “/sbin”
/mnt	The typical mount point for the user-mountable devices such external resources.

https://www.dba-oracle.com/linux/important_files_directories.htm
https://linuxhandbook.com/linux-directory-structure/
https://pagertree.com/blog/understanding-linux-file-system-a-comprehensive-guide-to-common-directories



Practice: What is the full path of the following files/directories?



1. sources.list
2. bash
3. fstab
4. tmp
5. Homework week 2.txt

Exploring subdirectories of the root directory

- Essential navigating commands:
  - **`pwd`** : Print current directory
  - **`ls`** : List files
  - **`cd`** : Change directory

Exploring subdirectories of the root directory

: Type of pathnames

- There are two types of pathnames:
  - **Absolute** – The full path to the file or directories begins from the root (/).
  - **Relative** – A partial path that relates to the current working directory. Relative paths do not start from the root.

Exploring subdirectories of the root directory

: Special characters

- Special characters interpreted by the shell for filename expansion:

`~`	Your home directory
`.`	Current directory
`..`	Parent directory
`*`	Wildcard matching any filename
`?`	Wildcard matching any character
`TAB`	Try to complete (partially typed) command or filename

Practice: Navigating the file system

`pwd`	Print working (current) directory
`cd /usr/local`	Change directory to /usr/local/
`cd ~`	Change to home directory
`cd ..`	Change directory to parent directory
`cd /`	Change directory to the root
`ls -l, ls -la, ls -lah, ls -lR, ls -lS, ls -lt`	Listing files and directories according to input command options

Creating and removing directories

- Create directories:
  - `$ mkdir directory`
  - `$ mkdir directory{<start>..<finish>}`
- Remove directories:
  - Remove empty directories:
    - `$ rmdir directories`
    - `$ rm -d directories`
  - Remove a non-empty directory and its content.
    - `$ rm -r directories`
- Remove files:
  - Remove single file:
    - `$ rm file1`
  - Remove multiple files:
    - `$ rm file1 file2 ... fileN`

Creating new files

- Create new file using ”touch” command
  - `$ touch filename.txt`
  - `$ touch filename{1..10}.txt`
- Create new file using text editor.
  - Vim
  - Nano
  - gedit
  - emacs
  - etc.

Practice: Create and remove file and directory

- Run the following processes:
  - Create 2 directories name “week1, week2” using “mkdir” command
  - Create file “test.txt” in folder “week1”
  - Change to directory “week2”
  - Create “file1.txt” using ”touch” command
  - Create “fileA”, “fileB”, “fileC” using touch command
  - Try remove file using the following command:
    - $ rm fileA
    - $ rm file?
    - $ rm file*
  - Remove directory “week1”
  - Remove directory “week2”

Editing files using text editor

VIM and Nano

Using VIM

- Vim is not pre-installed with Ubuntu. Therefore, you need to install it yourself.

```
$ sudo apt update
$ sudo apt install vim
```

There are 4 modes:

1. Command Mode	The default mode upon starting Vim. You can switch to any mode from this state.
2. Insert Mode	This mode is used to edit the content of the file. Switch to insert mode by pressing **“i"**. Use the Esc key to switch back to command mode.
3. Command-line Mode	This mode is used for working with commands. Switch to this mode by pressing “**:**” in command mode.
4. Visual Mode	Use for running “find and replace” command. Switch to this mode by pressing “**v**” in command mode. Use the Esc key to switch back to command mode.

https://www.freecodecamp.org/news/vim-beginners-guide/

Using VIM

Command	Description
`vim [file]`	Open or create new file
`y`	copy selected file
`d`	cut selected file
`p`	paste copy/cut selected file after the cursor
`P`	paste copy/cut selected file before the cursor
`:w`	Save the file
`:q`	Close the file and quit Vim
`:wq`	Save and quit Vim
`:q!`	Quit without save the file
`:[range]s/{pattern}/{string}/[flags]`	Command syntax to find and replace a text
`u`	Undo
`CTRL + R`	Redo

https://www.freecodecamp.org/news/vim-beginners-guide/

Find and Replace Text in Vim

```
:[range]s/{pattern}/{string}/[flags]
```

- [range] : To indicate the range of file to find.
  - % To find and replace in all lines
  - 5,10 To find and replace between lines 5 to 10
  - . To represent the current line
  - $ To represent the last line of the file
- {pattern} : Indicates the pattern to find the text.
- {string} : the string to replace in the found text.
- [flags] : Additional flags (for example
  - c To confirm before replacing the text)
  - i Case-insensitive search

Ex. Find and replace "Hello" with "Hi" in the entire file

```
:%s/Hello/Hi/g
```


https://www.freecodecamp.org/news/vim-beginners-guide/

Using NANO

- How to install NANO?

```
$ sudo apt install Nano
```

- How to open or create the new file using Nano?

```
$ nano [file]
```

- ” ^ “ symbol means CTRL (control) key on keyboard
- In the nano program, the ALT key in Windows is equivalent to the ESC key on Mac.

https://linuxhint.com/nano-editor-beginner-guide/

Using Nano

Command	Description
^g or F1	Display all the keyboard shortcuts used in Nano.
^o	Save changes to the file, prompting for the file name before saving.
^s	Save changes to the current file.
^w	Find for a word in the file.
ALT + w	To continue search further for the same text
^\	Find and replace the word in the file
^k	Cut the line
^u	Paste the line / selected text
ALT + A	To mark the text you want to copy
ALT + 6	To copy the selected text to the clipboard
ALT + U	Undo
ALT + E	Redo

https://linuxhint.com/nano-editor-beginner-guide/

Copy files and directories

- The 'cp' command is used to copy files from one location to another.
- Basic syntax:
  - `$cp [options] source destination`
    - options = command option
    - Sourec = files/directories
    - Destination = destination directory
- Copy single file:
  - `$cp file1.txt /tmp/`
- Copy multiple files:
  - `$cp file1.txt file2.txt /tmp/`
- Copy directory:
  - `$cp -r folder1 folder2 /tmp/`

https://www.freecodecamp.org/news/the-linux-cp-command-how-to-copy-files-in-linux/#:~:text=You%20can%20copy%20files%20by,this%20using%20the%20cp%20command.

Moving files and directories

- The ‘mv' command is used to move or rename files or directories.
- Basic syntax:
  - `$ mv [options] source destination`
  - Common useful options:
    - `-f, --force = do not prompt before overwriting`
    - `-i, --interactive = prompt before overwrite`
    - `-u, --update = move only when the SOURCE file is newer than the destination file`
- Move files/directories:
  - `$ mv file.txt folder /tmp/`
- Rename:
  - `$ mv old_name new_name`

https://www.thegeekdiary.com/linux-command-line-basics-for-beginners-copy-and-move-files-and-directories-cp-and-mv-commands/

Practice: CP and MV command

- Create file1, file2, file3
- Copy all three files to folder “/tmp”
- Make directory “folder1” and copy the folder to ”/tmp”
- Move “file1” to ”/tmp” using “mv -f” option
- Move “file2” to “/tmp” using “mv -i” option
- Change filename “file2” to “file2.2” then repeat the previous step
- Open and edit “file3” using “nano”. Then try running the following command:
  - `$ mv -u /tmp/file3 ./`
  - `$ cat file3`
  - `$ mv -u file3 /tmp/`
  - `$ cat /tmp/file3`

How to view the contents of a file using the CLI?

- cat command: This command use to concatenate, display, and create file in terminal. The 'cat' command will display the entire content of a file.
  - `$ cat filename.txt`
- less command: This command will display the content of the file one page at a time.
  - `$ less filename.txt`
- head command: This command will display the first 10 lines of the file by default. The option '-n' is used to specify the number of lines to display.
  - `$ head –n N filename.txt`
- tail command: This command will display the last 10 lines of the file by default. The option '-n' is used to specify the number of lines to display from the bottom.
  - `$ tail –n N filename.txt`
- more command: This command will display the content of the file one screen at a time.
  - `$ more filename.txt`
- nl command: This command will display the entire content of the file, similar to the 'cat' command. However, this command will also display line numbers alongside the contents.
  - `$ nl filename.txt`

https://unstop.com/blog/how-to-open-a-file-in-linux

Some useful file commands

`cp [file1] [file2]`	copy file
`mkdir [directory]`	make directory
`rmdir [directory]`	remove empty directory
`mv [file] [destination]`	move/rename file
`rm [file]`	remove (-r for recursive)
`file [file]`	identify file type
`more / less [file]`	page through file
`head -n` ***N*** `[file]`	display first ***N*** line
`tail -n` ***N*** `[file]`	display last ***N*** line
`cat [file1] [file2]`	Display or concatenate files
`tac [file1] [file2]`	Display or concatenate files in reverse order
`touch [file]` `touch filename{<start>..<finish>}`	Create new file or update modification time

Filtering text using the “grep” command

The 'grep' command can be used to search for a string or regular expression in a text file.

- Basic syntax:

```
$ grep "string" filename
```

- Colorizing Grep results using the --color option

```
$ grep --color "string" filename
```

- Searching for a string recursively in all directories

```
$ grep -r "string" *
```

- Ignoring case sensitivity

```
$ grep -i "string" filename
```

- Count the lines where strings are matched with -c option

```
$ grep -c "string" filename
```

- Using Grep to invert Output

```
$ grep -v "string" filename
```

- Number the lines that contain the search pattern with -n option

```
$ grep -n "string" filename
```

- Search for exact matching word using the -w option

```
$ grep -w "string" filename
```

- Search by regular expression using the -E option

```
$ grep -E “regex" filename
```


https://www.digitalocean.com/community/tutorials/grep-command-in-linux-unix

Practical: Grep command

- Try running various 'grep' command options on the following file.
  - gene.txt
  - primer.txt

Soft VS Hard links

- A link is a connection between a filename and the actual data. Multiple filenames can link to the same actual data. There are two types of links: soft links and hard links.
  - Soft Link or Symbolic links
    - A soft link points to the original file, similar to a shortcut in Windows. When the original file is deleted, the soft link will be broken.

```
$ ln -s  [original filename] [link name]
```

  - Hard link
    - A hard link creates a new file that points to the same actual data as the original file. The deletion of the original file will not affect the hard link.

```
$ ln  [original filename] [link name]
```

  - Changes in a hard link or soft link will update the original file.

Practice: Create Hard and Soft links

- Create hard link of “/tmp/file3”
- Create soft link of the previous hard link file.
- Run ”ls -la” command
- Update soft link file using “nano”
- Check content of original file (/tmp/file3) and hard link file.
- Remove file “/tmp/file3”, then check the hard link and soft link file.
- Remove hard link file, then check the soft link file.