Final Notes

1) Git and Markdown

A. How to clone a GitHub repository

Cloning a repository means copying a remote GitHub project to your local computer.

Example: `git clone https://github.com/username/repository.git`

B. How to use the git commands

Git Command Overview

Command	What It Does	When to Use It	Example
`git pull`	Downloads changes from a remote repository and merges them into your local branch.	Use this before starting work to make sure your local copy is up to date.	`git pull origin main`
`git add`	Stages files so Git knows which changes you want to include in the next commit.	Use this after modifying files and before committing.	`git add file.txt` `git add .`
`git commit`	Saves the staged changes to the local repository with a descriptive message.	Use this after staging files to record your changes.	`git commit -m "Updated README"`
`git push`	Uploads your local commits to a remote repository (such as GitHub).	Use this after committing to share your changes with others.	`git push origin main`

Typical Git Workflow

```
git pull
git add .
git commit -m "Describe your changes"
git push
```

C. How to write a Markdown file that contains images and proper formatting

Markdown Summary Table

Feature	Syntax / Example	Description

Feature	Syntax / Example	Description
Headings	`# Heading 1` `## Heading 2`	Create titles and subtitles
Bold Text	`**Bold**`	Make text bold
Italic Text	`*Italic*`	Make text italic
Strikethrough	`~~Text~~`	Strike through text
Unordered List	`- Item`	Create bullet points
Ordered List	`1. Item`	Create numbered lists
Links	`[GitHub](https://github.com)`	Add hyperlinks
Images	`![Alt text](image.png)`	Display images
Inline Code	`` `code` ``	Show code inline
Code Block	`bash ls -la `	Show multi-line code blocks
Blockquote	`> Quote`	Highlight quotes or notes
Horizontal Line	`---`	Add a dividing line

D. How to convert a Markdown file to PDF

```
1. Open README.md in VS Code
2. Right-click → Markdown PDF: Export (pdf)
3. Check the generated README.pdf in the same folder
```

2) How to compress (zip) a directory/folder in Debian

First, make sure zip is installed on the OS

```
whereis zip
sudo apt update
sudo apt install zip
```

Example to zip:

```
Command line to zip a folder:

zip -r my_project.zip my_project/

* my_project/ → the folder you want to compress
* my_project.zip → the output zip file
```

3) What are Absolute paths and relative paths? provide examples with commands. For example,creating a file using an absolute path.

In Linux, paths specify the location of files and directories. There are two types: **absolute** and **relative**.

---

1. Absolute Path

- Specifies the **full path from the root directory** `/`.
- Always starts with `/`.
- Works regardless of the current working directory.

Example: Creating a file using an absolute path

```
touch /home/user/documents/file.txt
```

2. Relative Path

Specifies a location relative to the current directory.

Does not start with /.

Depends on the directory you are currently in.

```
cd /home/user
touch documents/file.txt
```

Path Type	Starts With	Works From Anywhere?	Example
Absolute Path	`/`	✔ Yes	`/home/user/docs/file.txt`
Relative Path	Not `/`	✖ Only relative to current directory	`docs/file.txt`

4) How to work with the manual pages (man command)?

The `man` command in Linux is used to **view the manual pages** for other commands. It provides detailed information about the command, its options, and usage examples.

Example:

```
man command_name
```

5) How to Parse (Search) for Specific Words in Manual Pages

Linux allows you to **search for specific words** inside manual pages using built-in search or by combining `man` with other commands like `grep`.

Searching Inside `man` Using `/`

Open the manual for a command:

```
man ls
```

Press / followed by the word you want to search:

```
/color
```

Press Enter to go to the first occurrence.

Press n to move to the next match and N to move to the previous match.

6) How to redirect output (>, >>, and |)

Symbol	Purpose	Example
`>`	Overwrite output to a file	`ls > files.txt`
`>>`	Append output to a file	`echo "Hello" >> files.txt`
`l`	Pipe output to another command	`ls l grep ".txt"`

7) How to Append the Output of a Command to a File

In Linux, you can **append the output of a command** to an existing file using the `>>` operator.
This **adds the new output at the end of the file** without overwriting existing content.

Syntax:

```
command >> filename
```

7) How and When to Redirect Output Using Pipes (|)

In Linux, the **pipe operator** `|` allows you to **send the output of one command as input to another command**.
This is useful when you want to **process or filter data step by step**.

Syntax:

```
command1 | command2
```

8) How to use echo and output redirection to create a new file that contains some text

Creating a File with Text Using `echo` and Output Redirection

You can use the `echo` command along with `>` or `>>` to **create a new file containing text** in Linux.

Syntax

```
echo "Your text here" > filename.txt
```

9) How to use wildcards (For copying and moving multiple files at the same time)

Wildcards are symbols that **match one or more files** in Linux.
They are useful for copying or moving multiple files at the same time.

1. Common Wildcards

Wildcard	Meaning	Example
`*`	Matches any number of characters	`*.txt` → all `.txt` files
`?`	Matches exactly one character	`file?.txt` → `file1.txt` or `fileA.txt`
`[ ]`	Matches any one character inside brackets	`file[123].txt` → `file1.txt`, `file2.txt`, or `file3.txt`

2. Copy Multiple Files Using Wildcards

```
cp *.txt /home/user/backup/
```

Copies all .txt files from the current directory to /home/user/backup/.

```
cp file?.txt /home/user/backup/
```

3. Move Multiple Files Using Wildcards

```
mv *.log /home/user/logs/
```

Moves all .log files to /home/user/logs/.

```
mv report[1-3].pdf /home/user/reports/
```

10) How to use brace expansion (For creating entire directory structures in a single command)

Brace expansion `{}` in Linux allows you to **create multiple files or directories at once** without typing each name individually.

Syntax

```
mkdir parent_directory/{subdir1,subdir2,subdir3}
```

ExamplesL

- Example 1: Create multiple directories at once

```
mkdir projects/{project1,project2,project3}
```

Creates:

```
projects/project1
projects/project2
projects/project3
```

- Example 2: Create nested directories

```
mkdir -p projects/{project1,project2}/{src,bin,docs}
```

Creates the following structure:

```
projects/project1/src
projects/project1/bin
projects/project1/docs
projects/project2/src
projects/project2/bin
projects/project2/docs
```

11)How to create a simple “hello world” shell script

Script in txt editor:

```
#!/bin/bash
# This is a simple Hello World script
echo "Hello, World!"
```

run in console:

```
bash script.sh
```

Output:

```
Hello, World!
```

12)How to use variables in a shell script

```
#!/bin/bash
# Simple script using variables

name="Alice"
echo "Hello, $name!"
```

- No spaces before or after =
- Use $variable_name to access the value

13)For each of the following commands, include a definition, syntax/formula/usage/, and 2 - 5 well-documented examples.

Command	Definition	Syntax	Example
`awk`	Text processing tool for pattern scanning and processing	`awk 'pattern { action }' filename`	`awk '{print $1}' file.txt`
`cat`	Display or concatenate file content	`cat [options] filename`	`cat file.txt`
`cp`	Copy files or directories	`cp [options] source destination`	`cp file.txt /home/user/backup/`
`cut`	Extract sections from each line of a file	`cut -d 'delimiter' -f field_number filename`	`cut -d, -f1 data.csv`

Command	Definition	Syntax	Example
`grep`	Search for patterns in files	`grep [options] pattern filename`	`grep "error" log.txt`
`head`	Display the first part of a file	`head [options] filename`	`head -n 5 file.txt`
`ls`	List directory contents	`ls [options] [directory]`	`ls -l`
`man`	Display manual pages	`man command`	`man ls`
`mkdir`	Create new directories	`mkdir [options] directory_name`	`mkdir myfolder`
`mv`	Move or rename files/directories	`mv [options] source destination`	`mv oldname.txt newname.txt`
`tac`	Display file contents in reverse	`tac filename`	`tac file.txt`
`tail`	Display the last part of a file	`tail [options] filename`	`tail -n 10 file.txt`
`touch`	Create empty files or update timestamps	`touch filename`	`touch newfile.txt`
`tr`	Translate or delete characters	`tr [options] set1 set2`	`echo "hello" | tr 'a-z' 'A-Z'`
`tree`	Display directory structure in tree format	`tree [options] [directory]`	`tree -L 2`