**Part B**

Identify True or False:

1. ls is used to list files and directories in a directory.

Ans: True

The ls command in Linux is used to **list files and directories** in a specified directory. If no directory is specified, it lists the contents of the **current working directory**.

| **Command** | **Description** |
| --- | --- |
| ls | Lists files and directories in the current directory. |
| ls -l | Shows detailed information (permissions, owner, size, date). |
| ls -a | Includes **hidden files** (files starting with .). |
| ls -lh | Shows file sizes in **human-readable format** (KB, MB, GB). |
| ls -lt | Sorts by **modification time** (newest first). |

1. mv is used to move files and directories.

Ans: True

The mv command in Linux is used to **move** or **rename** files and directories.

| **Command** | **Description** |
| --- | --- |
| mv file1 file2 | Renames file1 to file2. |
| mv file /destination/ | Moves a file to a directory. |
| mv -i file1 file2 | **Interactive mode** (asks before overwriting). |
| mv -v file1 file2 | **Verbose mode** (shows what is being moved). |
| mv -u file1 /destination/ | Moves only if the **destination file is older**. |

1. cd is used to copy files and directories.

Ans: false

The cd command in Linux is **not** used for copying files and directories. It is used to **change directories**.

| **Command** | **Description** |
| --- | --- |
| cd /path/to/dir | Change to a specific directory (/path/to/dir). |
| cd | Moves to the **home directory** (/home/username). |
| cd ~ | Same as cd, moves to the **home directory**. |
| cd .. | Moves **one level up** in the directory tree. |
| cd ../.. | Moves **two levels up**. |
| cd - | Switches back to the **previous directory**. |
| cd / | Moves to the **root (/) directory**. |
| cd ~/Documents | Moves to the **Documents** folder in the user's home directory. |
| cd "$OLDPWD" | Same as cd -, moves to the **previous directory**. |

1. pwd stands for "print working directory" and displays the current directory.

Ans: True

The pwd command stands for **"Print Working Directory"** and is used to display the **absolute path** of the current working directory in Linux.

| **Option** | **Description** |
| --- | --- |
| pwd | Prints the **current working directory**. |
| pwd -L | Prints the logical (default) directory, including any symbolic links. |
| pwd -P | Prints the **physical path**, resolving symbolic links. |

1. grep is used to search for patterns in files.

Ans: True

The grep (**Global Regular Expression Print**) command in Linux is used to **search for patterns in files or text**

| **Option** | **Description** |
| --- | --- |
| grep "pattern" file.txt | Searches for "pattern" in file.txt. |
| grep -i "pattern" file.txt | Case-insensitive search. |
| grep -v "pattern" file.txt | Prints lines **NOT** containing "pattern". |
| grep -c "pattern" file.txt | Counts occurrences of "pattern" in the file. |
| grep -n "pattern" file.txt | Shows **line numbers** where "pattern" appears. |
| grep -r "pattern" /path | Recursively searches "pattern" in all files under /path. |
| grep -l "pattern" \*.txt | Lists filenames **that contain** "pattern". |
| grep -w "word" file.txt | Matches **whole words only**. |
| `grep -E "pattern1 | pattern2"` |
| grep -A 3 "pattern" file.txt | Shows **3 lines after** the matching line. |
| grep -B 3 "pattern" file.txt | Shows **3 lines before** the matching line. |
| grep -C 3 "pattern" file.txt | Shows **3 lines before & after** the match. |

1. chmod 755 file.txt gives read, write, and execute permissions to the owner, and read and execute permissions to group and others.

Ans: True

The command **chmod 755 file.txt** sets **file permissions** as follows:

* **Owner (7)** → **Read (r), Write (w), Execute (x)**
* **Group (5)** → **Read (r), Execute (x)**
* **Others (5)** → **Read (r), Execute (x)**

**Understanding chmod 755 file.txt**

| **User** | **Permissions (755)** | **Meaning** |
| --- | --- | --- |
| **Owner** | 7 = rwx | **Read, Write, Execute** |
| **Group** | 5 = r-x | **Read & Execute** (No Write) |
| **Others** | 5 = r-x | **Read & Execute** (No Write) |

**🔹 Explanation of Permission Digits**

Each number represents a **combination of permissions**:

| **Number** | **Binary** | **Permissions** |
| --- | --- | --- |
| 7 | 111 | rwx (Read, Write, Execute) |
| 5 | 101 | r-x (Read, Execute) |
| 0 | 000 | No Permissions |

1. mkdir -p directory1/directory2 creates nested directories, creating directory2 inside directory1 if directory1 does not exist.

he command **mkdir -p directory1/directory2** **creates nested directories**, meaning:

* It creates directory1 **if it does not already exist**.
* It then creates directory2 **inside directory1**.

**Explanation of mkdir -p directory1/directory2**

| **Command** | | **Description** | |
| --- | --- | --- | --- |
| mkdir directory1/directory2 | | ❌ **Fails** if directory1 does not exist. | |
| mkdir -p directory1/directory2 | | ✅ **Creates both directory1 and directory2** if they do not exist. | |
| **Option** | **Description** | | **Example** |
| mkdir directory\_name | Creates a **single directory**. | | mkdir myfolder |
| mkdir -p parent/child | Creates **nested directories** (does not fail if parent does not exist). | | mkdir -p project/src |
| mkdir -m 755 mydir | Creates a directory with **specific permissions**. | | mkdir -m 700 secure\_folder |
| mkdir -v mydir | **Verbose mode** (shows messages for each created directory). | | mkdir -v logs |
| mkdir -pv parent/child | **Combines -p and -v** (creates parent directories & shows messages). | | mkdir -pv backup/data |

1. rm -rf file.txt deletes a file forcefully without confirmation.

Ans False

The command **rm -rf file.txt** is **not correct for deleting a file** because the -r (**recursive**) option is meant for **directories**, not regular files.

**Correct Explanation of rm Command Options**

| **Option** | **Description** | **Correct Usage** |
| --- | --- | --- |
| rm file.txt | Deletes file.txt (asks for confirmation if write-protected). | rm myfile.txt |
| rm -f file.txt | **Force delete** file (file.txt) **without confirmation**. | rm -f myfile.txt |
| rm -r folder/ | Deletes a **directory and its contents recursively**. | rm -r myfolder/ |
| rm -rf folder/ | **Force delete** directory without confirmation. | rm -rf myfolder/ |

**Identify the Incorrect Commands:**

1. chmodx is used to change file permissions.

The correct command to change file permissions in Linux is **chmod**, **not** chmodx.

| **Command** | **Effect** |
| --- | --- |
| chmod 755 file.txt | **Owner: rwx**, **Group & Others: r-x** (Executable for all). |
| chmod 644 file.txt | **Owner: rw-**, **Group & Others: r--** (Read-only for others). |
| chmod +x script.sh | Makes script.sh **executable**. |
| chmod u+x file.txt | **Adds execute permission for the owner**. |

1. cpy is used to copy files and directories.

* There is **no command called** cpy in Linux. The correct command to **copy files and directories** is cp.

| **Task** | **Correct Command** |
| --- | --- |
| Copy a file | cp file1.txt file2.txt |
| Copy multiple files | cp file1.txt file2.txt /backup/ |
| Copy a directory | cp -r mydir/ /backup/ |
| Copy a file with confirmation | cp -i file.txt /backup/ |
| Copy a file and keep permissions | cp -p file.txt /backup/ |
| Copy only newer files | cp -u file.txt /backup/ |

1. mkfile is used to create a new file.

There is **no mkfile command in Linux** to create a new file. The correct commands to **create a new file** are:

| **Task** | **Correct Command** | **Description** |
| --- | --- | --- |
| Create an empty file | touch file.txt | **Creates a new empty file** (if it doesn’t exist). |
| Create and write text | echo "Hello" > file.txt | **Creates a file** and writes "Hello" to it. |
| Create a file using cat | cat > file.txt | **Creates a file** and allows typing content. |
| Create a file using nano | nano file.txt | Opens the file in the **Nano editor**. |
| Create a file using vim | vim file.txt | Opens the file in **Vim editor**. |

1. catx is used to concatenate files

There is **no catx command** in Linux. The correct command to **concatenate files** is **cat**.

| **Task** | **Correct Command** | **Description** |
| --- | --- | --- |
| Display file content | cat file.txt | Shows the contents of file.txt. |
| Concatenate multiple files | cat file1.txt file2.txt | Merges and displays file1.txt & file2.txt. |
| Save concatenated output | cat file1.txt file2.txt > merged.txt | Merges content into merged.txt. |
| Append to a file | cat file1.txt >> file2.txt | **Adds** file1.txt content to file2.txt. |

1. rn is used to rename files

There is **no rn command** in Linux to rename files. The correct command to **rename files** is **mv**.

| **Task** | **Correct Command** | **Description** |
| --- | --- | --- |
| Rename a file | mv oldname.txt newname.txt | Renames oldname.txt to newname.txt. |
| Rename a directory | mv olddir newdir | Renames olddir to newdir. |
| Move and rename a file | mv file.txt /home/user/newfile.txt | Moves and renames file.txt inside /home/user/. |