

**Module Code & Module Title**

Level 5 – CT5052NP

Assessment Type

Logbook Seven

Semester

2023/24 Spring/Autumn

Student Name: Subanta Poudel

London Met ID: 20048736

College ID: NP04CP4S210115

Assignment Due Date: December 21, 2024

Assignment Submission Date: December 21, 2024

Submitted To: Prasant Adhikari

Word Count (Where Required):

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded

Table of Contents

Table of Figures	3
Introduction	4
Tasks and Steps	5
Task 1: Creating Directory Structures.....	5
Command:	5
Verification:	5
Task 2: Changing to the 1level3 Directory	6
Command:	6
To navigate back and move to another directory:	6
Task 3: Navigating Between Directories	7
From 1level3 to 2level3:	7
From 2level3 to 4level3:	7
Task 4: Creating a Text File in 1level3.....	8
Command:	8
Verification:	8
Task 5: Copying Files	9
Commands:.....	9
Verification:	9
Task 6: Moving Files.....	10
Command:	10
Task 7: Printing Text Using Echo/Printf	11
Command:	11
Task 8: Using the Is Command.....	12
Commands:.....	12
Task 9: Removing Directories.....	14
Command:	14
Task 10: Managing File Permissions	15
Displaying Permissions:	15
Removing Permissions:	15
Adding Permissions:	15
Verification:	15

Task 11: Managing Directory Permissions	16
Commands:.....	16
Verification:	16
Observations	18
Conclusion	19

Table of Figures

Figure 1: installing tree function	5
Figure 2: Creating Directories	6
Figure 3: using cd command	6
Figure 4: Navigating between directories	7
Figure 5: Creating a text file	8
Figure 6: copying the file	9
Figure 7: Moving the file	10
Figure 8: Using echo -e to print multiple lines	11
Figure 9: Using ls command.....	12
Figure 10: Using ls -R function	13
Figure 11: Removing Directories	14
Figure 12: Removing directories	14
Figure 13: Managing file permissions.....	15
Figure 14: Managing directory permissions.....	16
Figure 15: Installing man command	17
Figure 16: using man command to manually check	18

Introduction

This log documents the tasks performed as part of Workshop 7, which focused on practicing various UNIX utilities. These tasks included creating directory structures, navigating directories using relative paths, managing files, modifying permissions, and working with UNIX commands like `mkdir`, `cd`, `chmod`, `ls`, and more. The primary aim was to enhance proficiency in directory and file operations in a UNIX environment. Few of the commands used:

- **mkdir**: Used to create directories. The `-p` option allows creating parent directories as needed.
- **sudo**: Executes commands with superuser privileges, essential for installing packages or modifying protected files.
- **apt**: A package management command used for installing, updating, or removing software on Debian-based systems.
- **tree**: Displays a visual representation of directory structures. Installed using `sudo apt-get install tree`.
- **cd**: Changes the current working directory. The `..` symbol refers to the parent directory.
- **cat**: Displays or creates files and appends content when used with `>>`.
- **ls**: Lists files and directories with options like:
 - o `-a`: Shows all files, including hidden ones.
 - o `-d`: Displays directories without listing their contents.
 - o `-g`: Omits owner information.
 - o `-l`: Provides detailed information.
 - o `-R`: Recursively lists files and directories.
- **cp**: Copies files or directories. Can rename files during copying.
- **mv**: Moves files or directories and can rename them.
- **chmod**: Modifies file or directory permissions. Common flags include `+r`, `+w`, and `+x` to add read, write, and execute permissions.
- **echo**: Outputs text to the terminal. The `-e` option enables interpretation of backslash escapes.
- **rm**: Removes files or directories. The `-i` option prompts before each removal.
- **rmdir**: Removes empty directories only.

This comprehensive set of commands provided the foundation for the tasks performed in this workshop, detailed below.

Tasks and Steps

Task 1: Creating Directory Structures

Create a directory structure with mkdir using relative paths.

Command:

```
sudo apt-get install tree
```

Installed the tree function to visualize the directory structure.

```
$ mkdir -p W7/{W7-1/{1level3,2level3},W7-2/{3level3,4level3}}
```

Here, the -p option ensures that parent directories are created if they do not exist.

Verification:

```
$ tree W7
```

The tree command displays the directory structure in a hierarchical format.

```
kali@kali: ~  
File Actions Edit View Help  
❏(kali@kali)~  
$ sudo apt update  
[sudo] password for kali:  
Get:1 http://kali.download/kali kali-rolling InRelease [41.5 kB]  
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [20.3 MB]  
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [48.9 MB]  
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [110 kB]  
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [262 kB]  
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [196 kB]  
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [877 kB]  
Get:8 http://kali.download/kali kali-rolling/non-free-firmware amd64 Packages [10.6 kB]  
Get:9 http://kali.download/kali kali-rolling/non-free-firmware amd64 Contents (deb) [23.2 kB]  
Fetched 70.7 MB in 18s (3,861 kB/s)  
1905 packages can be upgraded. Run 'apt list --upgradable' to see them.  
❏(kali@kali)~  
$ sudo apt install  
Summary:  
Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 1905  
❏(kali@kali)~  
$ sudo apt install tree  
Upgrading:  
tree  
Summary:  
Upgrading: 1, Installing: 0, Removing: 0, Not Upgrading: 1904  
Download size: 59.4 kB  
Space needed: 7,168 B / 64.4 GB available  
Get:1 http://mirror.kku.ac.th/kali kali-rolling/main amd64 tree amd64 2.2.1-1 [59.4 kB]  
Fetched 59.4 kB in 7s (8,217 B/s)  
(Reading database ... 395765 files and directories currently installed.)  
Preparing to unpack .../tree_2.2.1-1_amd64.deb ...  
Unpacking tree (2.2.1-1) over (2.1.3-1) ...  
Setting up tree (2.2.1-1) ...  
Processing triggers for man-db (2.12.1-2) ...  
Processing triggers for kali-menu (2024.3.1) ...  
❏(kali@kali)~  
$ sudo apt-get install tree  
[sudo] password for kali:  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
tree is already the newest version (2.2.1-1).  
tree set to manually installed.  
0 upgraded, 0 newly installed, 0 to remove and 1904 not upgraded.  
❏(kali@kali)~  
$
```

Figure 1: installing tree function

```
(kali㉿kali)-[~]
$ mkdir -p W7/{W7-1/{1level3,2level3},W7-2/{3level3,4level3}}

(kali㉿kali)-[~]
$ tree w7
w7 [error opening dir]

0 directories, 0 files

(kali㉿kali)-[~]
$ tree W7
W7
├── W7-1
│   ├── 1level3
│   └── 2level3
└── W7-2
    ├── 3level3
    └── 4level3

7 directories, 0 files

(kali㉿kali)-[~]
$
```

Figure 2: Creating Directories

Task 2: Changing to the 1level3 Directory

Navigate to the 1level3 directory using the cd command with relative paths.

Command:

```
$ cd W7/W7-1/1level3/
```

To navigate back and move to another directory:

```
$ cd ../../W7-2/4level3/
```

```
(kali㉿kali)-[~]
$ cd W7/W7-1/1level3/
```

Figure 3: using cd command

Task 3: Navigating Between Directories

Practice changing directories using relative pathnames.

From 1level3 to 2level3:

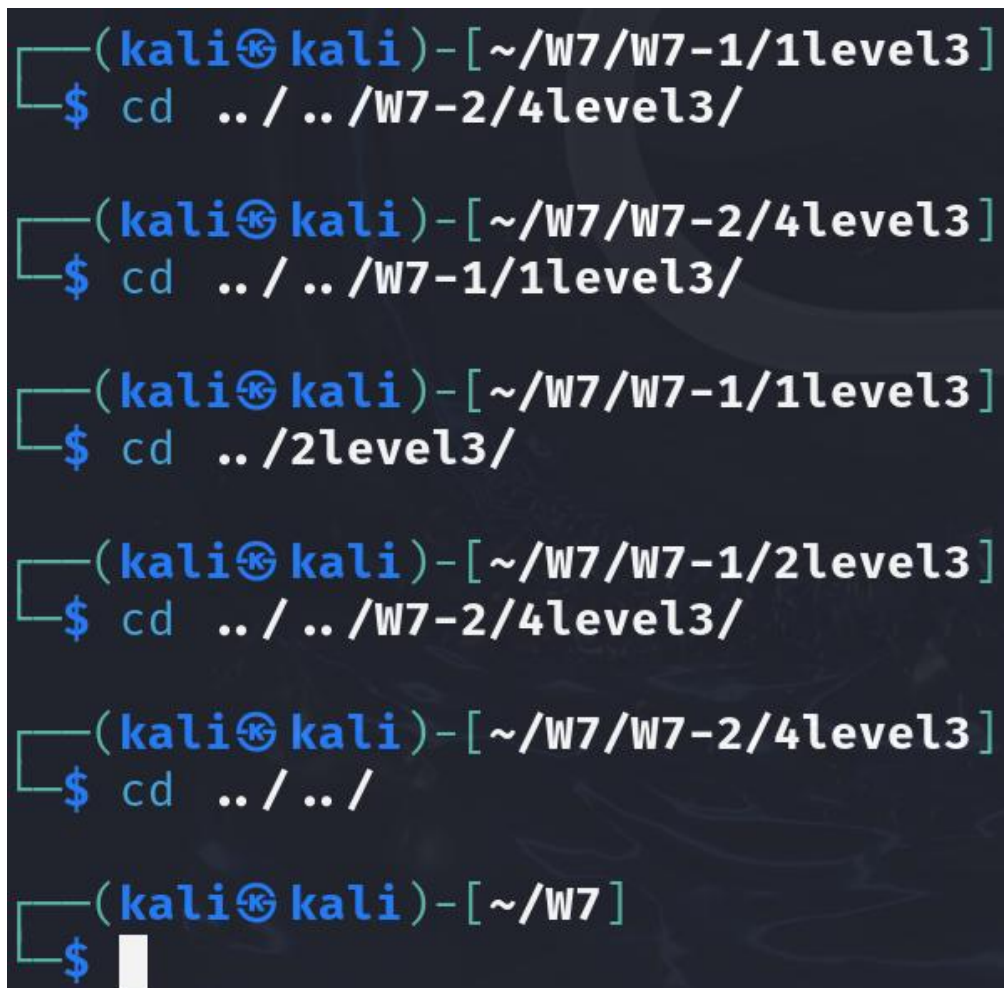
```
$ cd ../2level3/
```

From 2level3 to 4level3:

```
$ cd ../../W7-2/4level3/
```

Back to W7:

```
$ cd ../../
```



```
(kali㉿kali)-[~/W7/W7-1/1level3]
$ cd ../.. /W7-2/4level3/

(kali㉿kali)-[~/W7/W7-2/4level3]
$ cd ../.. /W7-1/1level3/

(kali㉿kali)-[~/W7/W7-1/1level3]
$ cd ../2level3/

(kali㉿kali)-[~/W7/W7-1/2level3]
$ cd ../.. /W7-2/4level3/

(kali㉿kali)-[~/W7/W7-2/4level3]
$ cd ../../

(kali㉿kali)-[~/W7]
$
```

Figure 4: Navigating between directories

Task 4: Creating a Text File in 1level3

We created a text file in 1level3 using the cat command.

Command:

```
$ W7-1/1level3/
```

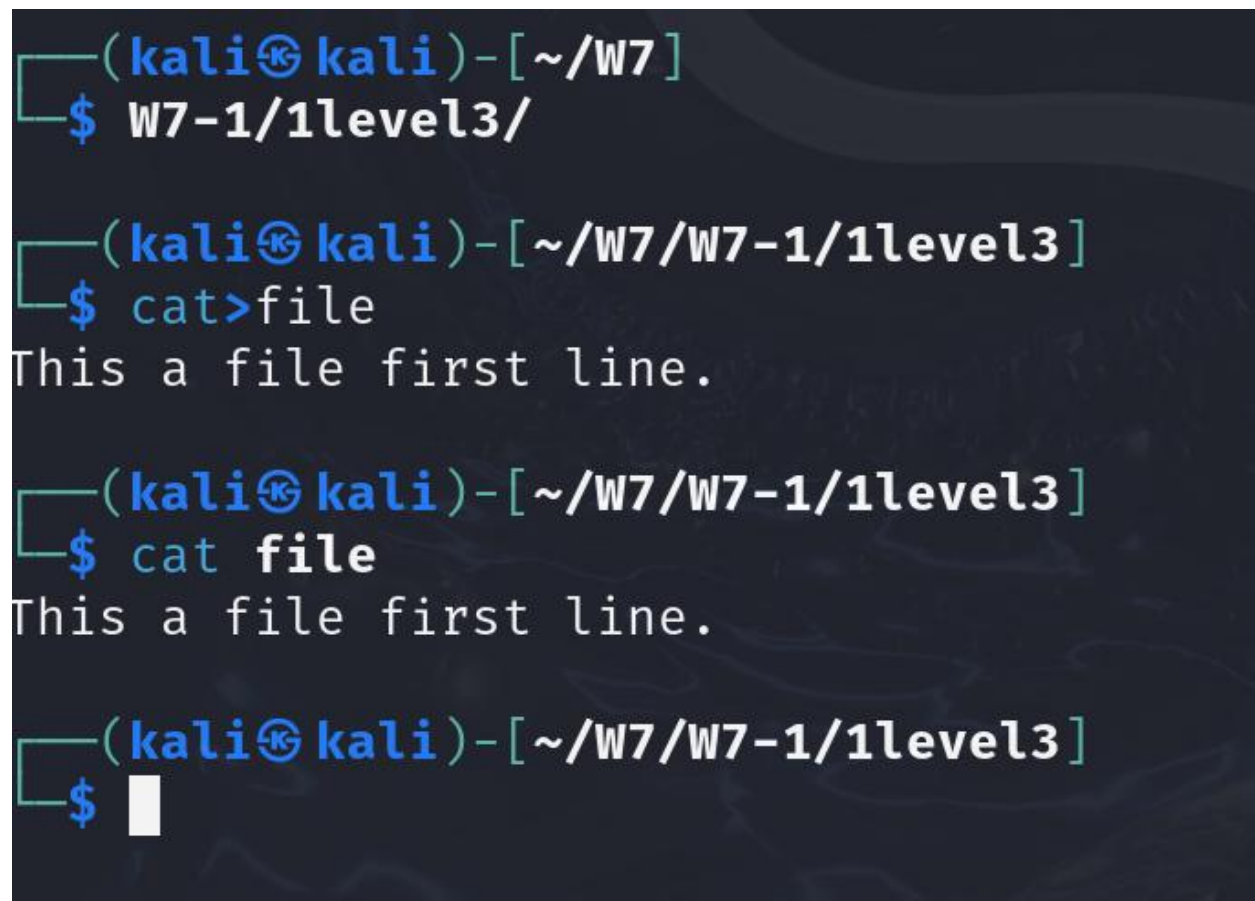
```
$ cat > file
```

```
This is a file.
```

```
Press Ctrl+D to save and exit.
```

Verification:

```
$ cat file
```

A terminal window with a dark background and light blue text. The prompt is (kali@kali)-[~/W7]. The user enters \$ W7-1/1level3/. The prompt changes to (kali@kali)-[~/W7/W7-1/1level3]. The user enters \$ cat>file. The text 'This a file first line.' is entered. The prompt changes back to (kali@kali)-[~/W7/W7-1/1level3]. The user enters \$ cat file. The text 'This a file first line.' is displayed. The prompt changes back to (kali@kali)-[~/W7/W7-1/1level3]. The user enters \$ followed by a white cursor block.

```
(kali@kali)-[~/W7]
$ W7-1/1level3/

(kali@kali)-[~/W7/W7-1/1level3]
$ cat>file
This a file first line.

(kali@kali)-[~/W7/W7-1/1level3]
$ cat file
This a file first line.

(kali@kali)-[~/W7/W7-1/1level3]
$
```

Figure 5: Creating a text file

Task 5: Copying Files

We copied the text file to other directories and renamed it during the copy operation.

Commands:

```
$ cp file file1
```

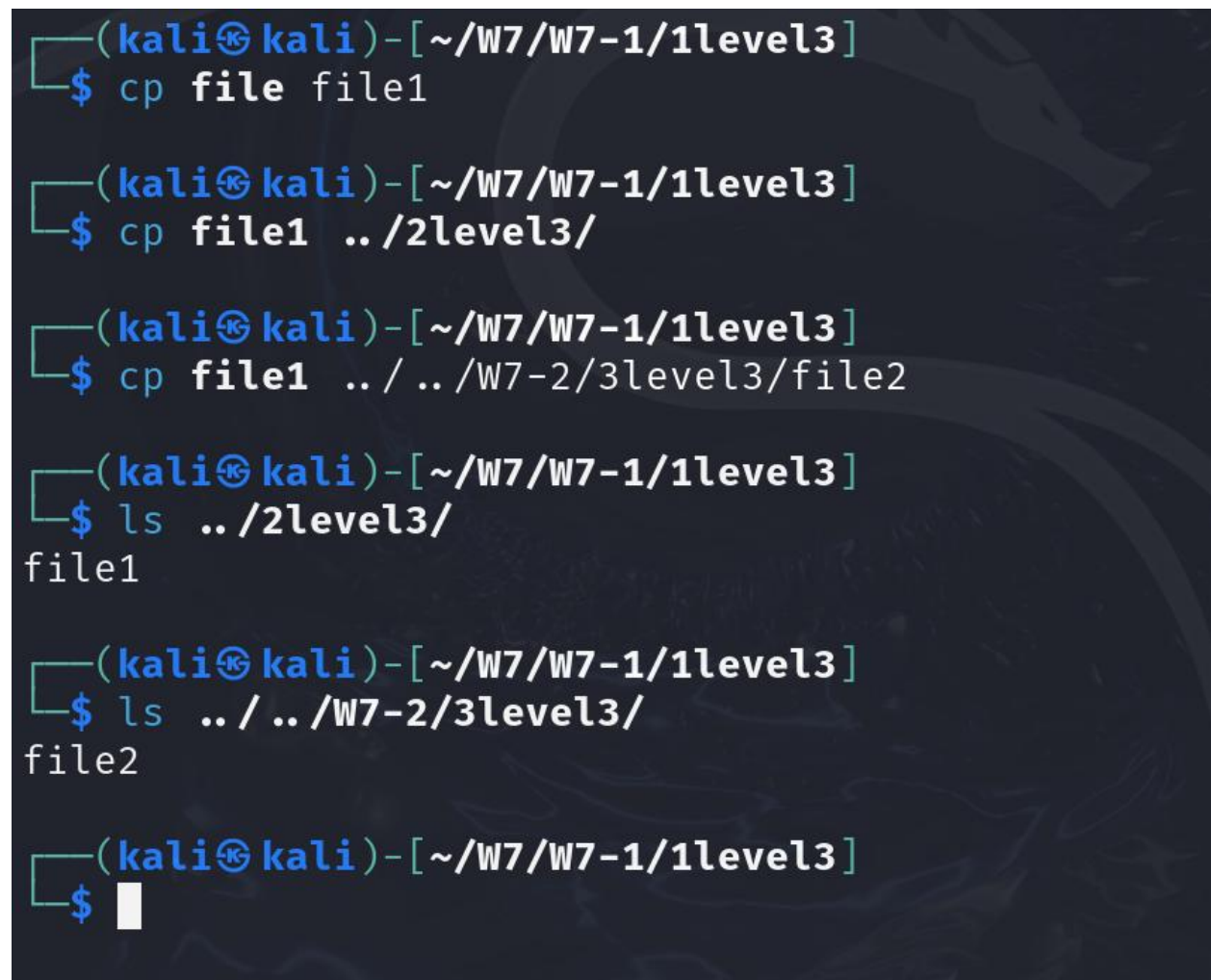
```
$ cp file1 ../2level3/
```

```
$ cp file1 ../../W7-2/3level3/renamed_file
```

Verification:

```
$ ls ../2level3/
```

```
$ ls ../../W7-2/3level3/
```

A terminal window with a dark background and a Kali Linux logo watermark. The prompt is (kali㉿kali)-[~/W7/W7-1/1level3]. The user enters several commands to copy a file named 'file' to different locations and then lists the contents of those locations to verify the copy. The commands and their outputs are as follows:

```
(kali㉿kali)-[~/W7/W7-1/1level3]
$ cp file file1

(kali㉿kali)-[~/W7/W7-1/1level3]
$ cp file1 ../2level3/

(kali㉿kali)-[~/W7/W7-1/1level3]
$ cp file1 ../../W7-2/3level3/file2

(kali㉿kali)-[~/W7/W7-1/1level3]
$ ls ../2level3/
file1

(kali㉿kali)-[~/W7/W7-1/1level3]
$ ls ../../W7-2/3level3/
file2

(kali㉿kali)-[~/W7/W7-1/1level3]
$
```

Figure 6: copying the file

Task 6: Moving Files

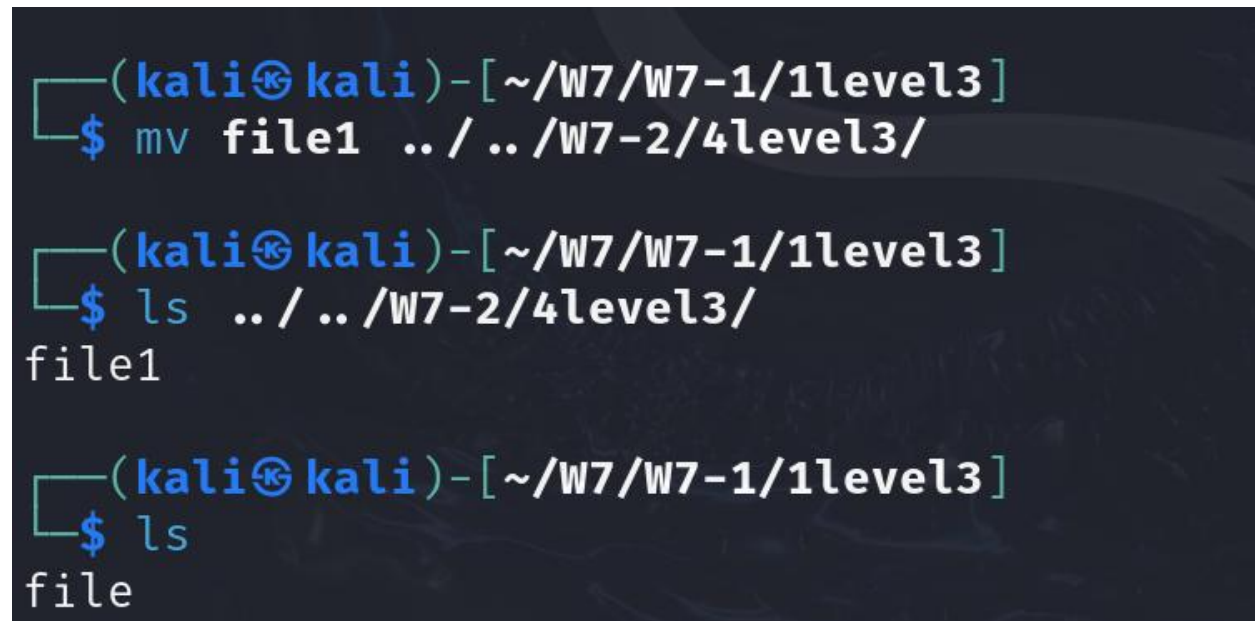
We moved file1 to 4level3 and verified its removal from 1level3.

Command:

```
$ mv file1 ../../W7-2/4level3/
```

```
$ ls ../../W7-2/4level3/
```

```
$ ls
```

A terminal window with a dark background and light blue/green text. The prompt is (kali@kali)-[~/W7/W7-1/1level3]. The user enters the command mv file1 ../../W7-2/4level3/. The prompt changes to (kali@kali)-[~/W7/W7-1/1level3]. The user enters the command ls ../../W7-2/4level3/. The output is file1. The prompt changes to (kali@kali)-[~/W7/W7-1/1level3]. The user enters the command ls. The output is file.

```
(kali@kali)-[~/W7/W7-1/1level3]  
$ mv file1 ../../W7-2/4level3/  
  
(kali@kali)-[~/W7/W7-1/1level3]  
$ ls ../../W7-2/4level3/  
file1  
  
(kali@kali)-[~/W7/W7-1/1level3]  
$ ls  
file
```

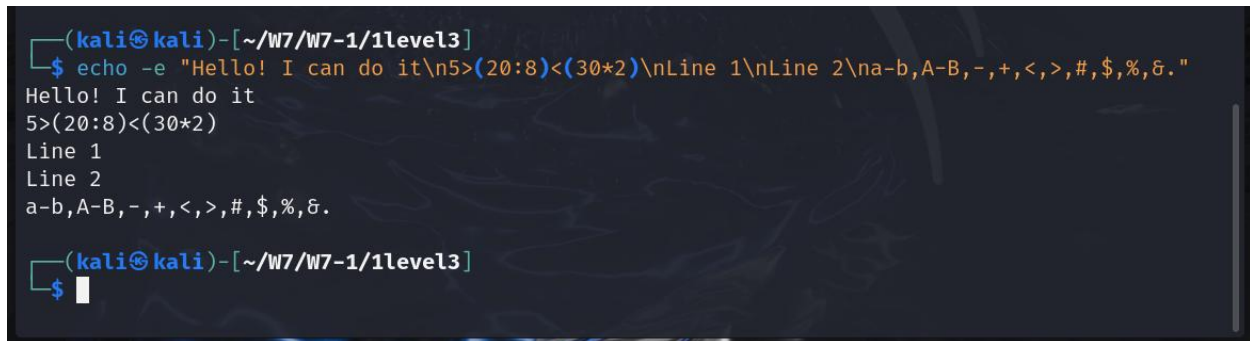
Figure 7: Moving the file

Task 7: Printing Text Using Echo/Printf

We printed multiple lines of text using echo with the -e option.

Command:

```
$ echo -e "Hello! I can do it\n5 > (20:8) < (30 * 2)\nLine 1\nLine 2\na-b, A-B, -, +, <, >, #,\n$, %, &."
```

A terminal window with a dark background and light blue text. The prompt is (kali@kali) - [~/W7/W7-1/1level3]. The command \$ echo -e "Hello! I can do it\n5 > (20:8) < (30 * 2)\nLine 1\nLine 2\na-b, A-B, -, +, <, >, #,\n\$, %, &." is entered. The output is displayed line by line: Hello! I can do it, 5 > (20:8) < (30 * 2), Line 1, Line 2, a-b, A-B, -, +, <, >, #, \$, %, &. The prompt returns to (kali@kali) - [~/W7/W7-1/1level3] with a cursor on the next line.

```
(kali@kali) - [~/W7/W7-1/1level3]
$ echo -e "Hello! I can do it\n5 > (20:8) < (30 * 2)\nLine 1\nLine 2\na-b, A-B, -, +, <, >, #,\n$, %, &."
Hello! I can do it
5 > (20:8) < (30 * 2)
Line 1
Line 2
a-b, A-B, -, +, <, >, #, $, %, &.
(kali@kali) - [~/W7/W7-1/1level3]
$
```

Figure 8: Using echo -e to print multiple lines

Task 8: Using the ls Command

We executed the ls command with various options to list contents.

Commands:

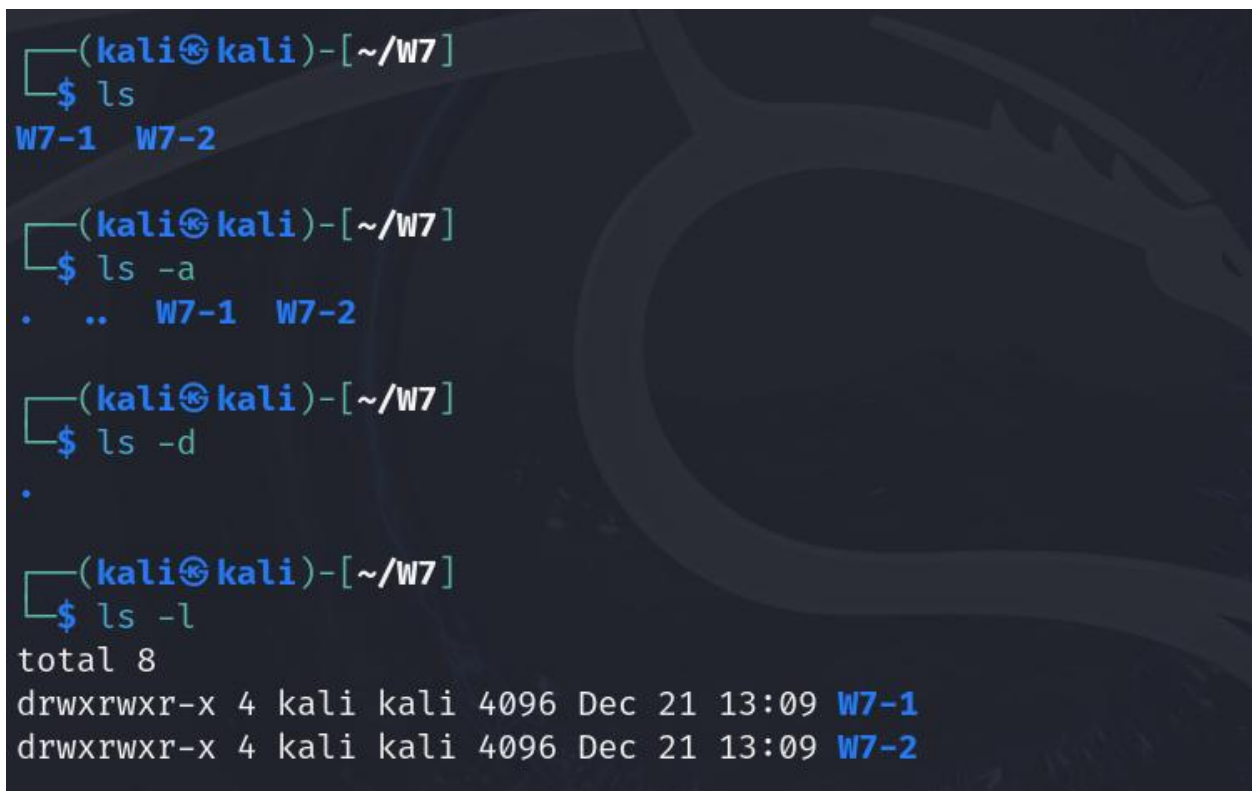
```
$ ls
```

```
$ ls -a
```

```
$ ls -d
```

```
$ ls -l
```

```
$ ls -R
```



```
(kali㉿kali)-[~/W7]
$ ls
W7-1  W7-2

(kali㉿kali)-[~/W7]
$ ls -a
.  ..  W7-1  W7-2

(kali㉿kali)-[~/W7]
$ ls -d
.

(kali㉿kali)-[~/W7]
$ ls -l
total 8
drwxrwxr-x 4 kali kali 4096 Dec 21 13:09 W7-1
drwxrwxr-x 4 kali kali 4096 Dec 21 13:09 W7-2
```

Figure 9: Using ls command

```
(kali㉿kali)-[~/W7]  
$ ls -R  
..  
W7-1  W7-2  
  
./W7-1:  
1level3  2level3  
  
./W7-1/1level3:  
file  
  
./W7-1/2level3:  
file1  
  
./W7-2:  
3level3  4level3  
  
./W7-2/3level3:  
file2  
  
./W7-2/4level3:  
file1  
  
(kali㉿kali)-[~/W7]  
$ █
```

Figure 10: Using ls -R function

Task 9: Removing Directories

We removed directories using `rm` and `rmdir` commands with the `-i` option for confirmation.

Command:

```
$ rm -ri W7-2/3level3/
```

```
$ rmdir W7-2/4level3/
```

```
(kali㉿kali)-[~/W7]
$ rm -ri W7-2/3level3/
rm: descend into directory 'W7-2/3level3/'?
```

Figure 11: Removing Directories

```
(kali㉿kali)-[~/W7/W7-2]
$ rm -ri W7-2/3level3/
rm: cannot remove 'W7-2/3level3/': No such file or directory

(kali㉿kali)-[~/W7/W7-2]
$ rmdir W7-2/4level3/
rmdir: failed to remove 'W7-2/4level3/': No such file or directory

(kali㉿kali)-[~/W7/W7-2]
$ cd ..

(kali㉿kali)-[~/W7]
$ rmdir W7-2/4level3/
rmdir: failed to remove 'W7-2/4level3/': Directory not empty

(kali㉿kali)-[~/W7]
$
```

Figure 12: Removing directories

Task 10: Managing File Permissions

We explored and modified file permissions using the chmod command.

Displaying Permissions:

```
$ ls -l 1level3/file
```

Removing Permissions:

```
$ chmod -rw 1level3/file
```

Adding Permissions:

```
$ chmod u+rw 1level3/file
```

Verification:

```
$ ls -l 1level3/file
```

```
(kali㉿kali)-[~/W7]
$ W7-1

(kali㉿kali)-[~/W7/W7-1]
$ ls -l 1level3/file
-rw-rw-r-- 1 kali kali 24 Dec 21 13:21 1level3/file

(kali㉿kali)-[~/W7/W7-1]
$ chmod -rw 1level3/file

(kali㉿kali)-[~/W7/W7-1]
$ ls -l 1level3/file
----- 1 kali kali 24 Dec 21 13:21 1level3/file

(kali㉿kali)-[~/W7/W7-1]
$ chmod u+rw 1level3/file

(kali㉿kali)-[~/W7/W7-1]
$ chmod u+rw 1level3/file

(kali㉿kali)-[~/W7/W7-1]
$ ls -l 1level3/file
-rw----- 1 kali kali 24 Dec 21 13:21 1level3/file

(kali㉿kali)-[~/W7/W7-1]
$
```

Figure 13: Managing file permissions

Task 11: Managing Directory Permissions

We performed operations to modify and verify directory permissions.

Commands:

```
$ chmod -rwx 1level3/
```

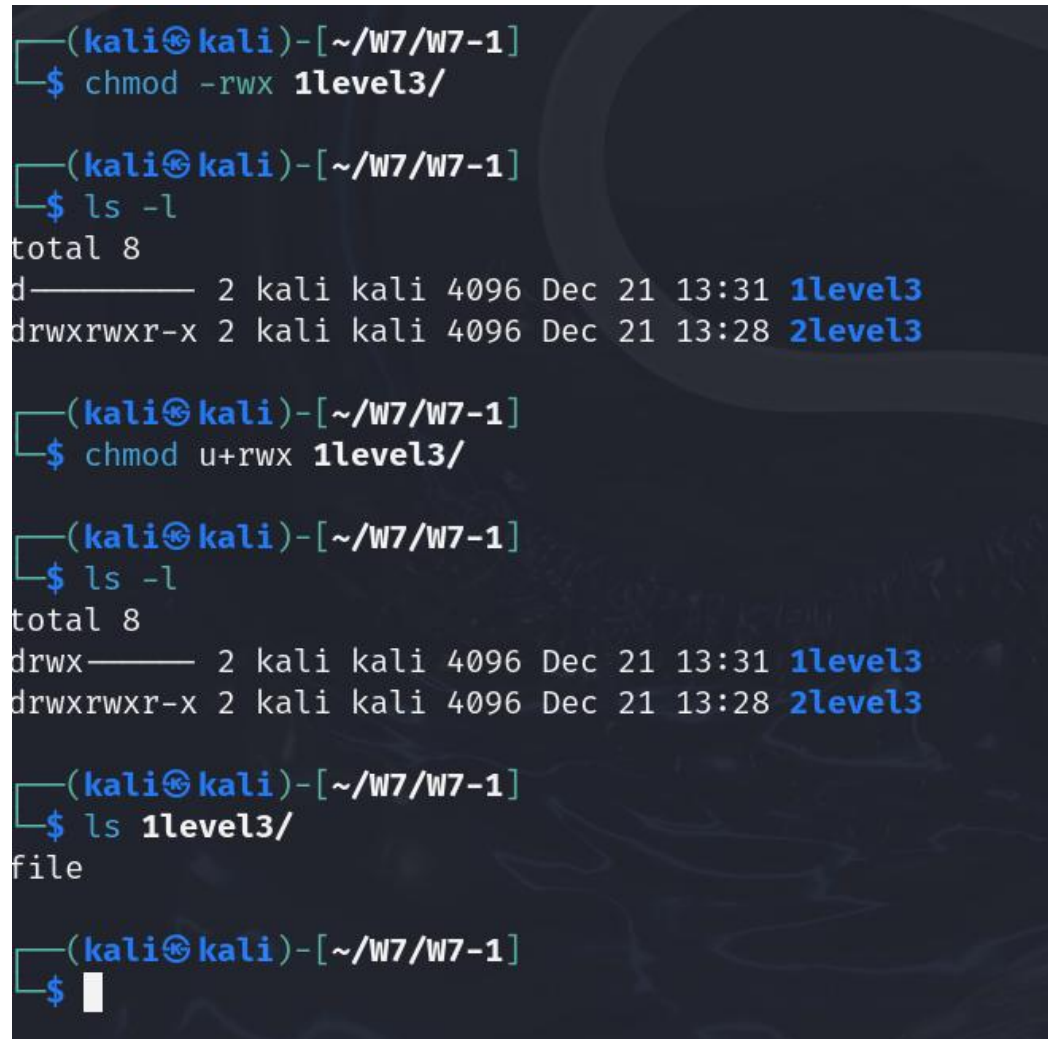
```
$ ls -l
```

```
$ chmod u+rwx 1level3/
```

```
$ ls -l
```

Verification:

```
$ ls 1level3/
```



```
(kali㉿kali)-[~/W7/W7-1]
└─$ chmod -rwx 1level3/

(kali㉿kali)-[~/W7/W7-1]
└─$ ls -l
total 8
d----- 2 kali kali 4096 Dec 21 13:31 1level3
drwxrwxr-x 2 kali kali 4096 Dec 21 13:28 2level3

(kali㉿kali)-[~/W7/W7-1]
└─$ chmod u+rwx 1level3/

(kali㉿kali)-[~/W7/W7-1]
└─$ ls -l
total 8
drwx----- 2 kali kali 4096 Dec 21 13:31 1level3
drwxrwxr-x 2 kali kali 4096 Dec 21 13:28 2level3

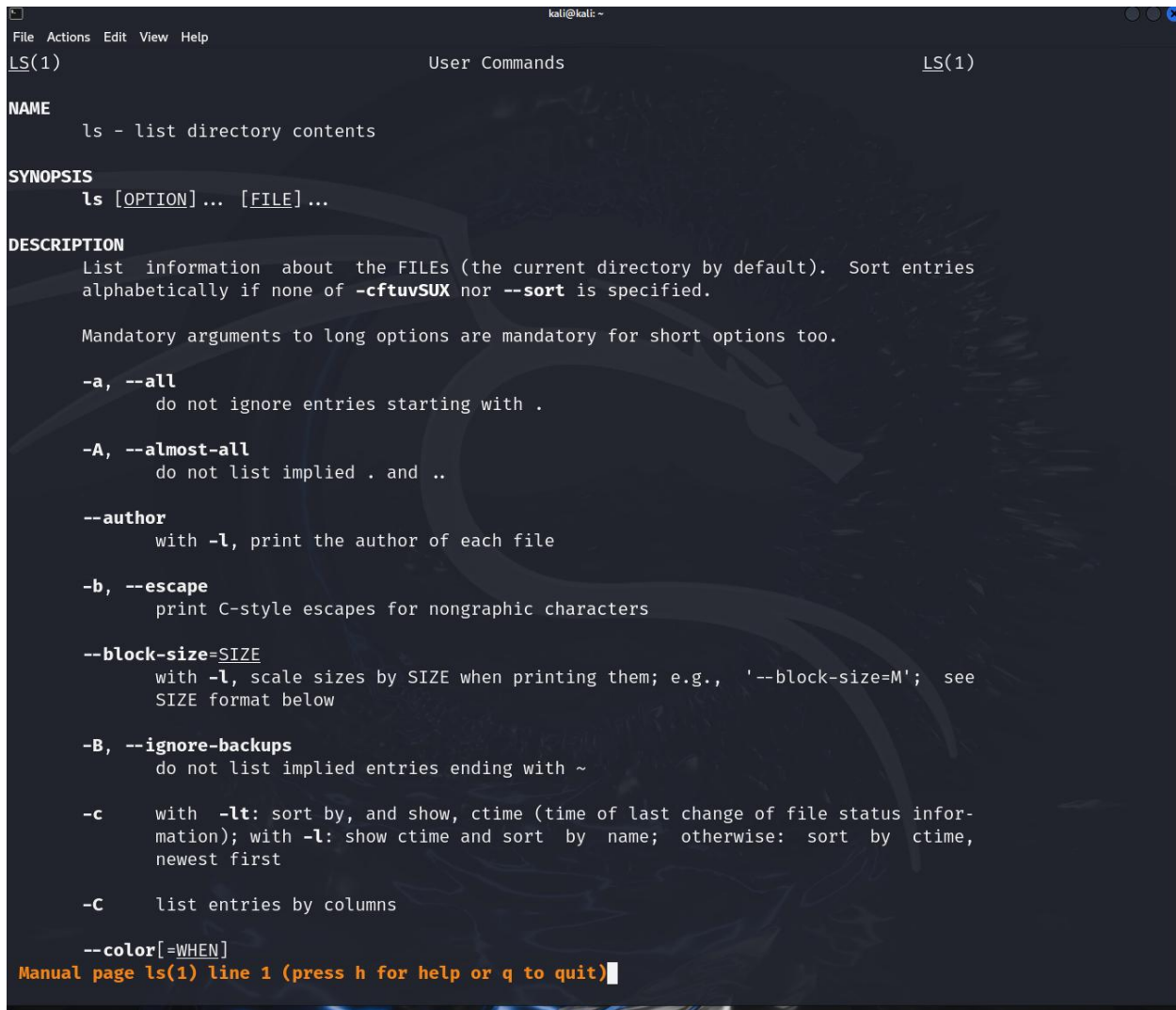
(kali㉿kali)-[~/W7/W7-1]
└─$ ls 1level3/
file

(kali㉿kali)-[~/W7/W7-1]
└─$
```

Figure 14: Managing directory permissions


```
(kali㉿kali)-[~/W7/W7-1]
$ sudo apt-get install man
[sudo] password for kali:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'man-db' instead of 'man'
The following packages will be upgraded:
  man-db
1 upgraded, 0 newly installed, 0 to remove and 1903 not upgraded.
Need to get 1,420 kB of archives.
After this operation, 38.9 kB of additional disk space will be used.
Get:1 http://kali.download/kali kali-rolling/main amd64 man-db amd64 2.13.0-1 [1,420 kB]
Fetched 1,420 kB in 1s (2,184 kB/s)
Preconfiguring packages ...
(Reading database ... 395765 files and directories currently installed.)
Preparing to unpack .../man-db_2.13.0-1_amd64.deb ...
Unpacking man-db (2.13.0-1) over (2.12.1-2) ...
Setting up man-db (2.13.0-1) ...
Updating database of manual pages ...
man-db.service is a disabled or a static unit not running, not starting it.
Processing triggers for kali-menu (2024.3.1) ...
Processing triggers for doc-base (0.11.2) ...
Processing 40 changed doc-base files...
Processing triggers for mailcap (3.72) ...
```

Figure 15: Installing man command



```
kali@kali: ~
File Actions Edit View Help
LS(1) User Commands LS(1)

NAME
    ls - list directory contents

SYNOPSIS
    ls [OPTION]... [FILE]...

DESCRIPTION
    List information about the FILES (the current directory by default). Sort entries
    alphabetically if none of -cftuvSUX nor --sort is specified.

    Mandatory arguments to long options are mandatory for short options too.

    -a, --all
        do not ignore entries starting with .

    -A, --almost-all
        do not list implied . and ..

    --author
        with -l, print the author of each file

    -b, --escape
        print C-style escapes for nongraphic characters

    --block-size=SIZE
        with -l, scale sizes by SIZE when printing them; e.g., '--block-size=M'; see
        SIZE format below

    -B, --ignore-backups
        do not list implied entries ending with ~

    -c
        with -lt: sort by, and show, ctime (time of last change of file status infor-
        mation); with -l: show ctime and sort by name; otherwise: sort by ctime,
        newest first

    -C
        list entries by columns

    --color[=WHEN]
Manual page ls(1) line 1 (press h for help or q to quit)
```

Figure 16: using man command to manually check

Observations

1. Basic Linux commands like mkdir, cd, rm, and cat were successfully used to manage files and directories.
2. The outputs of ls, ls -a, and ls -al were compared to understand hidden files and file permissions.
3. The tree command proved useful for visualizing directory structures.
4. File and directory permissions were modified using chmod, demonstrating its flexibility in user access control.

Conclusion

In this log, we explored:

1. Core Linux commands for file and directory management.
2. Creating and managing directory structures efficiently using commands like `mkdir` and `tree`.
3. Modifying and verifying permissions for files and directories.
4. Utilizing commands such as `ls`, `cat`, and `chmod` to understand and control file access.

This workshop provided practical experience with essential UNIX commands and concepts like directory structures, relative paths, file management, and permissions. By executing these tasks, we strengthened our understanding of how to effectively navigate and manipulate the UNIX file system.