M4 Lab: File System (Part 2)

CITA 171: OPERATING SYSTEM USE & ADMIN

Gonzalez, Marco A.
SUNY CANTON | 34 CORNELL DRIVE, CANTON, NEW YORK 13617

TABLE OF CONTENTS

1 TABLE OF FIGURES

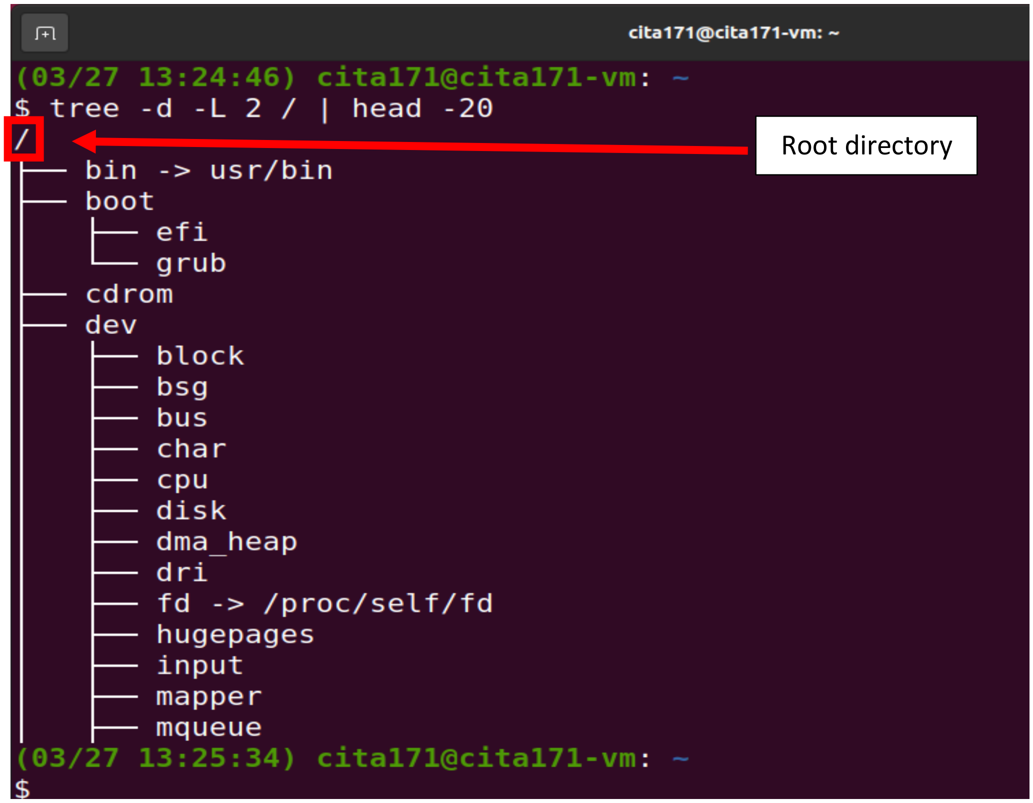
2 TABLE OF TABLES

3 PREPARATION

Start the CITA 171 VM and log in. Start a Terminal program.

4 DIRECTORY TREE

A file system organizes directories in the form of an inverted tree called a **directory tree**. Execute the following command. This command displays the directory tree structure starting from the root of the inverted tree called the **root directory**. The root directory is always represented with a slash character (/). See Figure 1.

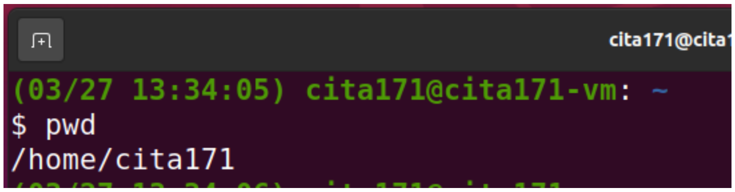


```
(03/27 13:24:46) cita171@cita171-vm: ~
$ tree -d -L 2 / | head -20
/
├── bin -> usr/bin
├── boot
│   ├── efi
│   └── grub
├── cdrom
├── dev
│   ├── block
│   ├── bsg
│   ├── bus
│   ├── char
│   ├── cpu
│   ├── disk
│   ├── dma_heap
│   ├── dri
│   ├── fd -> /proc/self/fd
│   ├── hugepages
│   ├── input
│   ├── mapper
│   ├── mqueue
(03/27 13:25:34) cita171@cita171-vm: ~
$
```

Figure 1. Directory Tree (resembling an inverted tree)

5 Finding the Current Working Directory

The **pwd** command displays the current directory path called the **current working directory**. See Figure 2.



```
(03/27 13:34:05) cita171@cita171-vm: ~
$ pwd
/home/cita171
```

Figure 2. The pwd Command

The displayed path **/home/cita171** is in the format called an **absolute path**. See Table 1.

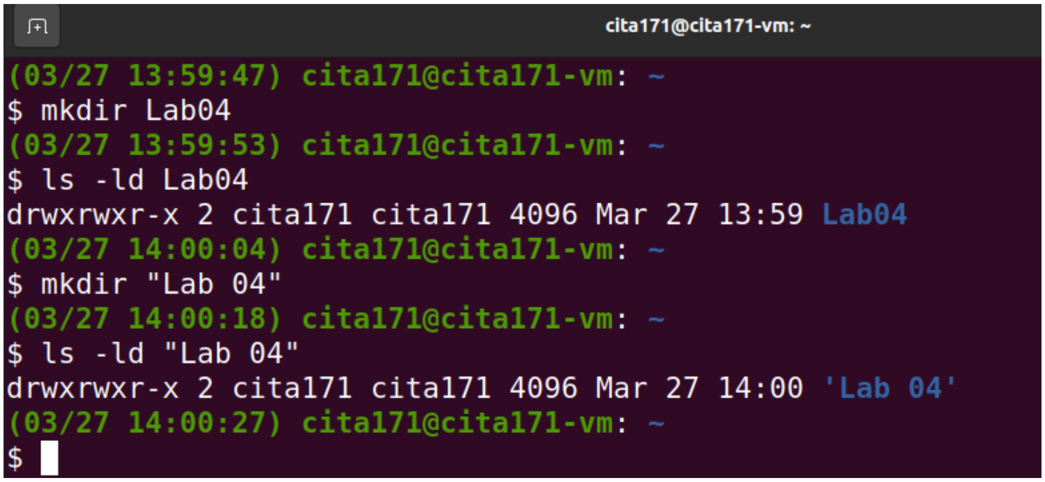
Table 1. Absolute Path Structure

/home/cita171		
/	home	cita171
Root Directory	Subdirectory of Root Directory	Subdirectory of home

The default directory when the user logs in is called the **user's home directory**. Therefore, /home/cita171 is cita171 user's home directory. The symbol tilde (~) is a shorthand notation to mean the user's home directory. Consequently, for the cita171 user, ~ means /home/cita171.

6 Creating a Directory

The **mkdir** command is used to create a new directory. If the directory name has spaces, it must be enclosed between double quotes. The **ls** command with the **-ld** options is used to check whether the directory exists or not. In this example, new directories name *Lab04* and *Lab 04* are created, and they are checked if they have been successfully created. See Figure 3.

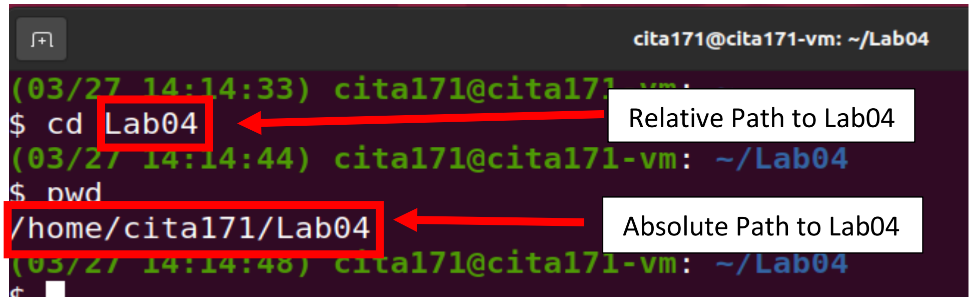


```
(03/27 13:59:47) cita171@cita171-vm: ~
$ mkdir Lab04
(03/27 13:59:53) cita171@cita171-vm: ~
$ ls -ld Lab04
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 13:59 Lab04
(03/27 14:00:04) cita171@cita171-vm: ~
$ mkdir "Lab 04"
(03/27 14:00:18) cita171@cita171-vm: ~
$ ls -ld "Lab 04"
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 14:00 'Lab 04'
(03/27 14:00:27) cita171@cita171-vm: ~
$ 
```

Figure 3. The mkdir Command

7 Changing the Directory

The **cd** command is used to change the current working directory to a different directory. It needs the path to the different directory as a command argument. The path can be either an absolute path or a **relative path**. In this example, the current working directory is changed from */home/cita171* to */home/cita171/Lab04* using a relative path, and then the new working directory is checked. See Figure 4



```
(03/27 14:14:33) cita171@cita171
$ cd Lab04
(03/27 14:14:44) cita171@cita171-vm: ~/Lab04
$ pwd
/home/cita171/Lab04
(03/27 14:14:48) cita171@cita171-vm: ~/Lab04
$ 
```

Figure 4. The cd Command

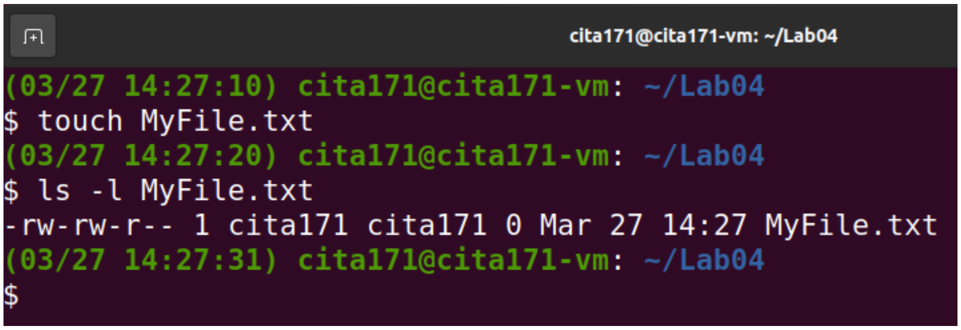
An absolute path always starts from the root directory (/). On the other hand, a relative path starts from the current working directory. In this case, the current working directory is (/home/cita171), as shown in Figure 2. Note that a relative path does not start with a slash (/). The current working directory is hidden (implied). See Table 2.

Table 2. Absolute vs. Relative Paths to /home/cita171/Lab04

	Current Working Directory	Subdirectory
Absolute Path	/home/cita171/	Lab04
Relative Path	(hidden, implied)	Lab04

8 CREATING A FILE

The **touch** command is used to create an empty (zero-byte) file. The **ls** command with the **-l** option is used to check if the file is successfully created. In this example, a new empty named *MyFile.txt* is created, and the ls command is used to check if the file has been successfully created. See Figure 5.

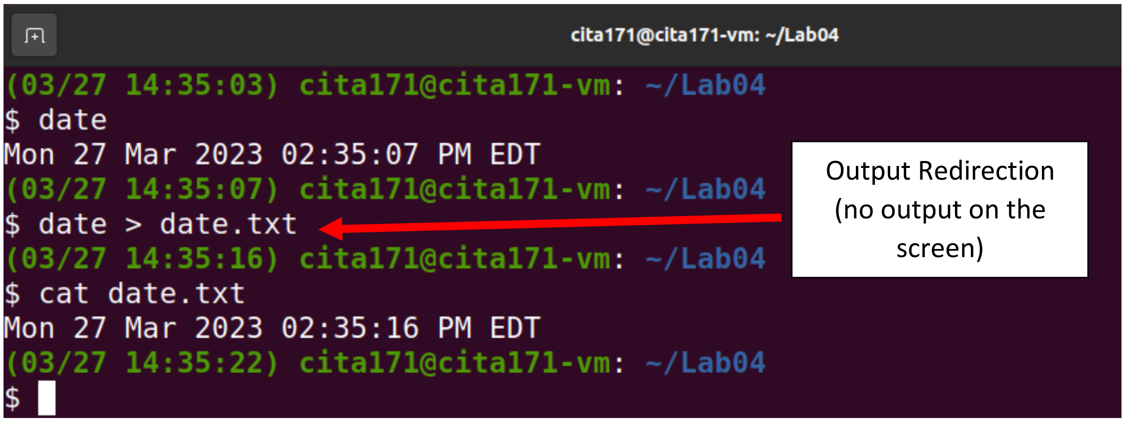


```
(03/27 14:27:10) cita171@cita171-vm: ~/Lab04
$ touch MyFile.txt
(03/27 14:27:20) cita171@cita171-vm: ~/Lab04
$ ls -l MyFile.txt
-rw-rw-r-- 1 cita171 cita171 0 Mar 27 14:27 MyFile.txt
(03/27 14:27:31) cita171@cita171-vm: ~/Lab04
$
```

Figure 5. The touch Command

9 CREATING A FILE WITH CONTENTS (OUTPUT REDIRECTION)

Output redirection is used to send command output to a different device instead of displaying it on the screen. A greater-than sign (>) is added at the end of the command, followed by the name of the target device. For example, if the target device is a regular file name, the command output is saved to the file (i.e., on a disk device). The file is called an **output file**. In this example, the date command's output is redirected (saved) to a file name *date.txt*, and the **cat** command is used to display the contents of the file. See Figure 6. Notice that the redirected data command does not display the output on the screen.



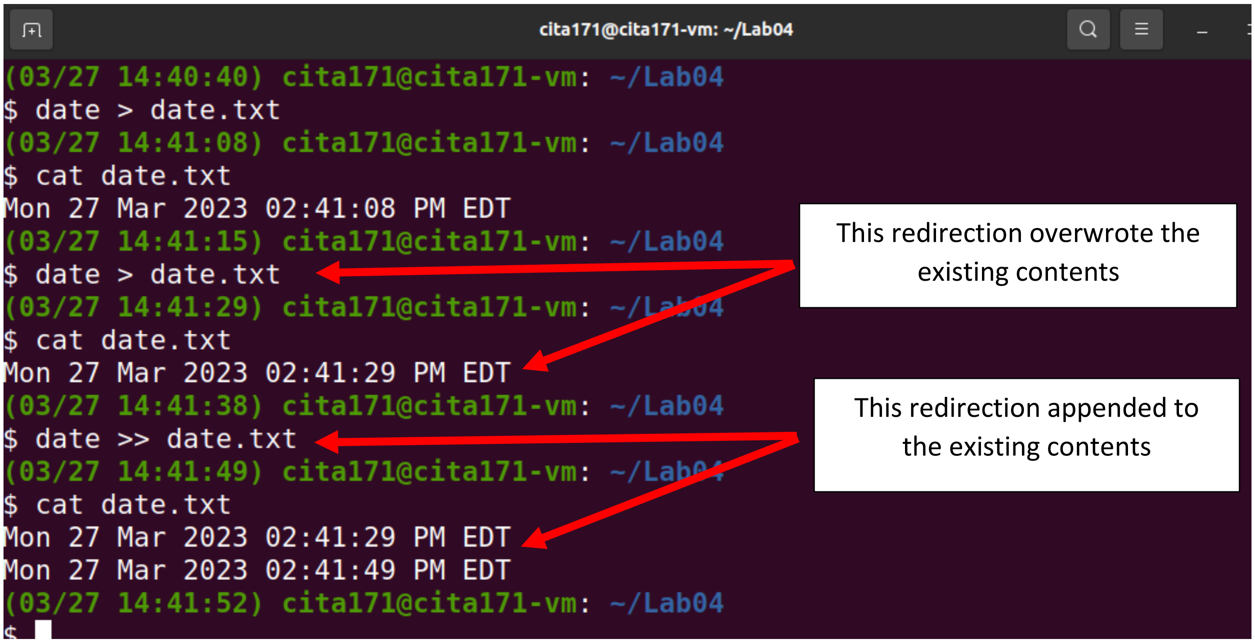
```
(03/27 14:35:03) cita171@cita171-vm: ~/Lab04
$ date
Mon 27 Mar 2023 02:35:07 PM EDT
(03/27 14:35:07) cita171@cita171-vm: ~/Lab04
$ date > date.txt
(03/27 14:35:16) cita171@cita171-vm: ~/Lab04
$ cat date.txt
Mon 27 Mar 2023 02:35:16 PM EDT
(03/27 14:35:22) cita171@cita171-vm: ~/Lab04
$ 
```

Output Redirection (no output on the screen)

Figure 6. Output Redirection

10 Adding More Contents to an Existing File

The greater-than sign (>) is an output redirection in the **overwrite mode**. If the redirected file already contains contents, they are overwritten. The two greater-than signs (>>) are an output redirection in the **append mode**. If the redirected file already contains contents, the new contents are added at the end of the existing contents. The date command is executed in the overwrite and append modes to show the differences in this example. See Figure 7.



```
(03/27 14:40:40) cita171@cita171-vm: ~/Lab04
$ date > date.txt
(03/27 14:41:08) cita171@cita171-vm: ~/Lab04
$ cat date.txt
Mon 27 Mar 2023 02:41:08 PM EDT
(03/27 14:41:15) cita171@cita171-vm: ~/Lab04
$ date > date.txt
(03/27 14:41:29) cita171@cita171-vm: ~/Lab04
$ cat date.txt
Mon 27 Mar 2023 02:41:29 PM EDT
(03/27 14:41:38) cita171@cita171-vm: ~/Lab04
$ date >> date.txt
(03/27 14:41:49) cita171@cita171-vm: ~/Lab04
$ cat date.txt
Mon 27 Mar 2023 02:41:29 PM EDT
Mon 27 Mar 2023 02:41:49 PM EDT
(03/27 14:41:52) cita171@cita171-vm: ~/Lab04
$ 
```

This redirection overwrote the existing contents

This redirection appended to the existing contents

Figure 7. Overwrite and Append Output Redirections

11 Reading the Output of the LS -L Command

The ls command with the -l option displays detailed information about the files and directories. Consider the ls command output in Figure 5. The output shows eight pieces of information. See Table 3.

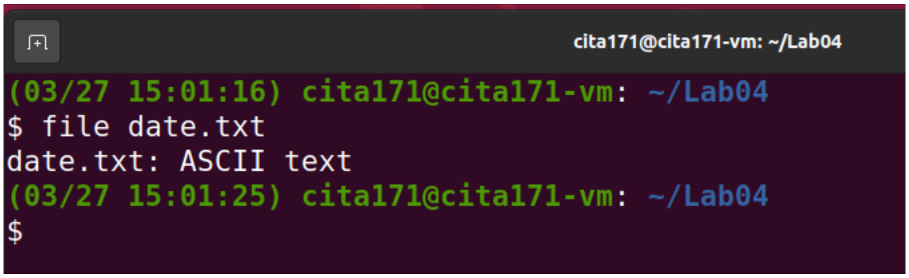
Table 3. The ls -l Command Output Structure

1	2	3	4	5	6	7	8
-	rw-rw-rw--	1	cita171	cita171	0	Oct 24 11:23	MyFile.txt
Object Type	Permissions	Hard Link Count	Owner	Group	Size (bytes)	Last Modified Date	Object Name

The object type is a **minus sign** (-) if the object is a **regular file**. It is a letter **d** if the object is a directoyy. Note that when the object type is a regular file, the ls command cannot identify exactly what file type it is (e.g., Word document, PDF document, photo).

12 Identifying the Exact Object Type

The **file** command is used to identify the exact object type. See Figure 8. The command correctly identified as a text file.

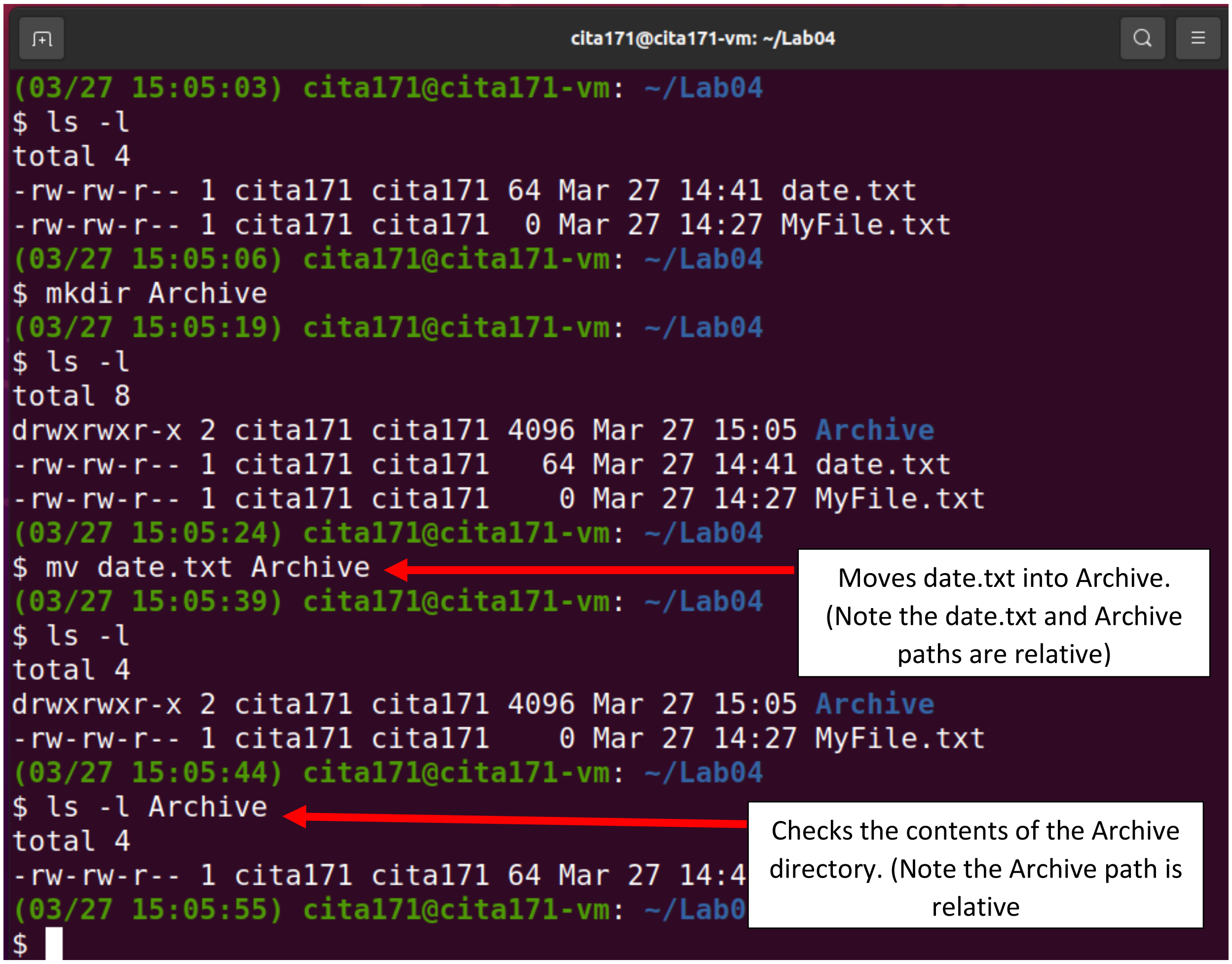


```
(03/27 15:01:16) cita171@cita171-vm: ~/Lab04
$ file date.txt
date.txt: ASCII text
(03/27 15:01:25) cita171@cita171-vm: ~/Lab04
$
```

Figure 8. The file Command

13 Moving a File/Directory

The **mv** command is used to relocate a file or a directory to a different file system location. In this example a new directory called *Archive* is created, and *date.txt* is moved into the directory. Then the ls command is used to check if the file has been successfully moved. See Figure 9.



```
(03/27 15:05:03) cita171@cita171-vm: ~/Lab04
$ ls -l
total 4
-rw-rw-r-- 1 cita171 cita171 64 Mar 27 14:41 date.txt
-rw-rw-r-- 1 cita171 cita171  0 Mar 27 14:27 MyFile.txt
(03/27 15:05:06) cita171@cita171-vm: ~/Lab04
$ mkdir Archive
(03/27 15:05:19) cita171@cita171-vm: ~/Lab04
$ ls -l
total 8
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 15:05 Archive
-rw-rw-r-- 1 cita171 cita171   64 Mar 27 14:41 date.txt
-rw-rw-r-- 1 cita171 cita171    0 Mar 27 14:27 MyFile.txt
(03/27 15:05:24) cita171@cita171-vm: ~/Lab04
$ mv date.txt Archive
(03/27 15:05:39) cita171@cita171-vm: ~/Lab04
$ ls -l
total 4
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 15:05 Archive
-rw-rw-r-- 1 cita171 cita171    0 Mar 27 14:27 MyFile.txt
(03/27 15:05:44) cita171@cita171-vm: ~/Lab04
$ ls -l Archive
total 4
-rw-rw-r-- 1 cita171 cita171 64 Mar 27 14:4
(03/27 15:05:55) cita171@cita171-vm: ~/Lab0
$ 
```

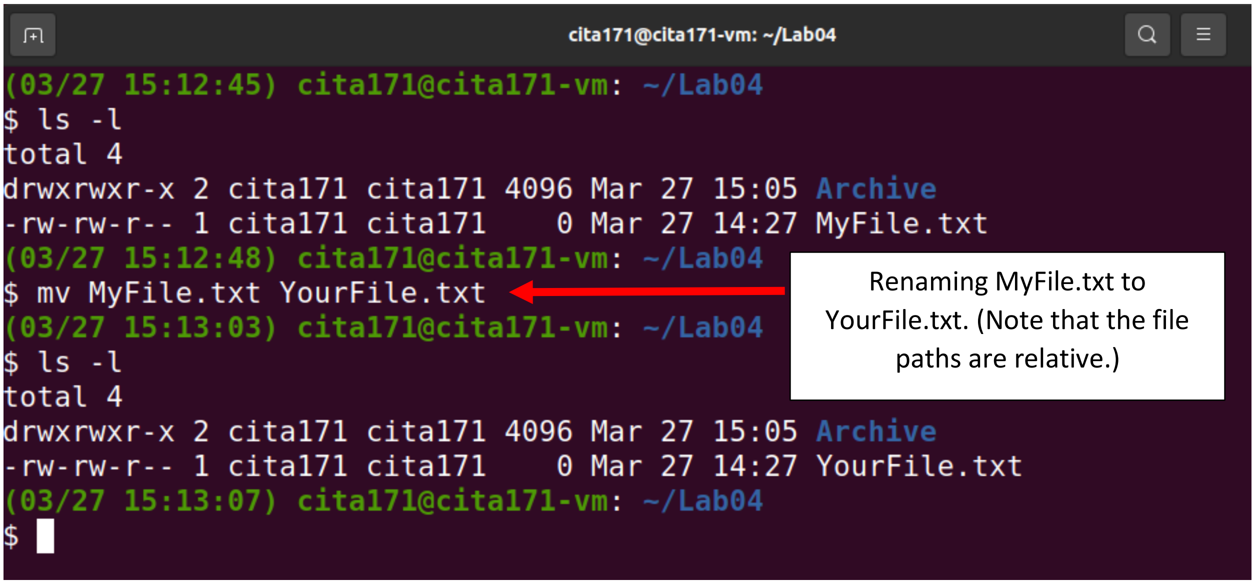
Moves date.txt into Archive. (Note the date.txt and Archive paths are relative)

Checks the contents of the Archive directory. (Note the Archive path is relative

Figure 9. The mv Command to Relocate a File

14 Renaming a File/Directory

The **mv** command is also used to rename a file or a directory. In this example, *MyFile.txt* is renamed as *YourFile.txt*. Then the ls command is used to check if the renaming has been successful. See Figure 10.



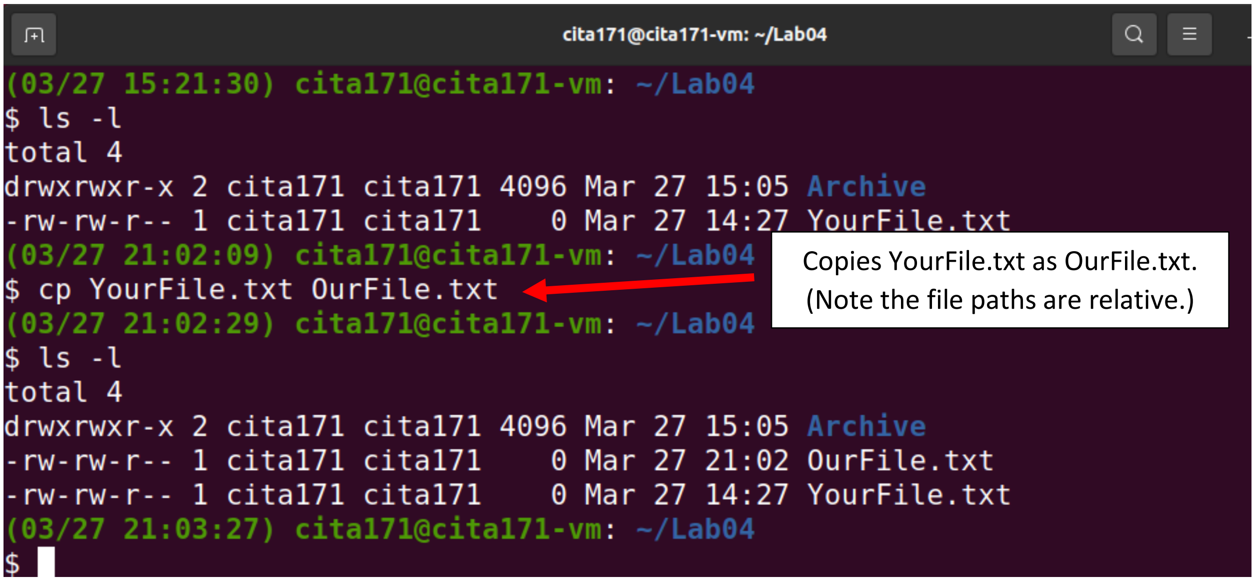
```
(03/27 15:12:45) cita171@cita171-vm: ~/Lab04
$ ls -l
total 4
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 15:05 Archive
-rw-rw-r-- 1 cita171 cita171    0 Mar 27 14:27 MyFile.txt
(03/27 15:12:48) cita171@cita171-vm: ~/Lab04
$ mv MyFile.txt YourFile.txt
(03/27 15:13:03) cita171@cita171-vm: ~/Lab04
$ ls -l
total 4
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 15:05 Archive
-rw-rw-r-- 1 cita171 cita171    0 Mar 27 14:27 YourFile.txt
(03/27 15:13:07) cita171@cita171-vm: ~/Lab04
$ 
```

Renaming MyFile.txt to YourFile.txt. (Note that the file paths are relative.)

Figure 10. The mv Command to Rename a File

15 COPYING A FILE

The **cp** command is used to copy a file. In this example, a copy *YourFile.txt* is created as *OurFile.txt*. Then the ls command is used to check if the copying has been successful. See Figure 11.



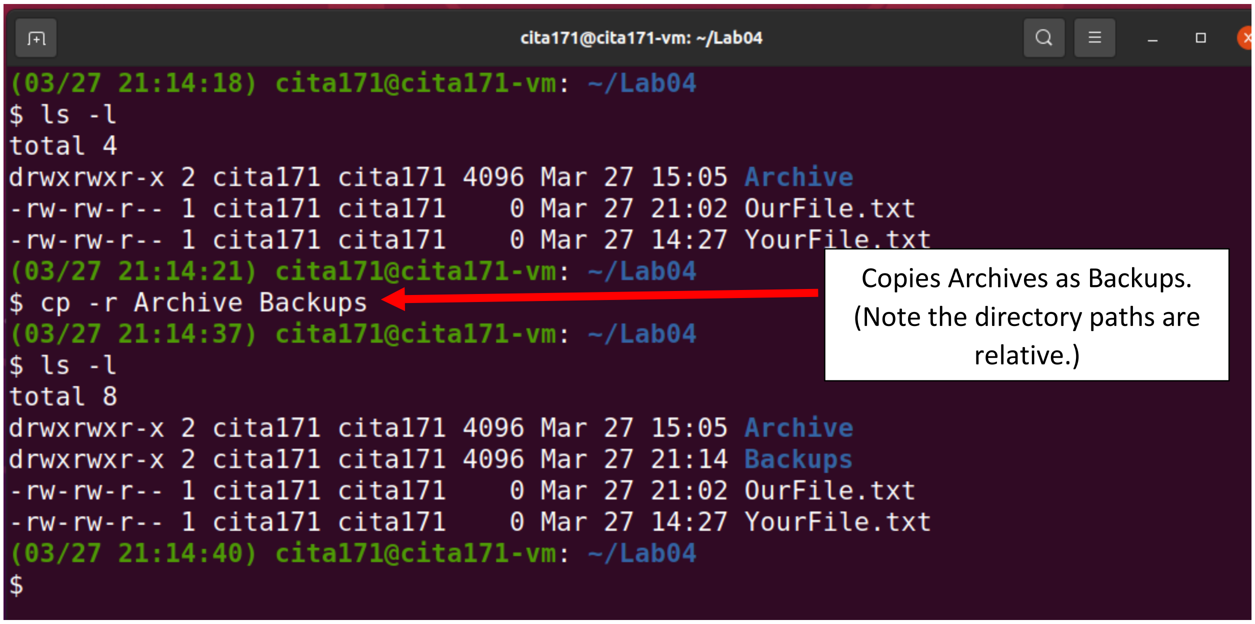
```
(03/27 15:21:30) cita171@cita171-vm: ~/Lab04
$ ls -l
total 4
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 15:05 Archive
-rw-rw-r-- 1 cita171 cita171    0 Mar 27 14:27 YourFile.txt
(03/27 21:02:09) cita171@cita171-vm: ~/Lab04
$ cp YourFile.txt OurFile.txt
(03/27 21:02:29) cita171@cita171-vm: ~/Lab04
$ ls -l
total 4
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 15:05 Archive
-rw-rw-r-- 1 cita171 cita171    0 Mar 27 21:02 OurFile.txt
-rw-rw-r-- 1 cita171 cita171    0 Mar 27 14:27 YourFile.txt
(03/27 21:03:27) cita171@cita171-vm: ~/Lab04
$ 
```

Copies YourFile.txt as OurFile.txt. (Note the file paths are relative.)

Figure 11. The cp Command to Copy a File

16 Copying a Directory

The **cp** command is also used to copy a directory; however, the **-r** (recursive copy) option must be added. In this example, a copy of the *Archive* directory is coped as *Backups*. Then the ls command is used to check if the copying has been successful. See Figure 12.



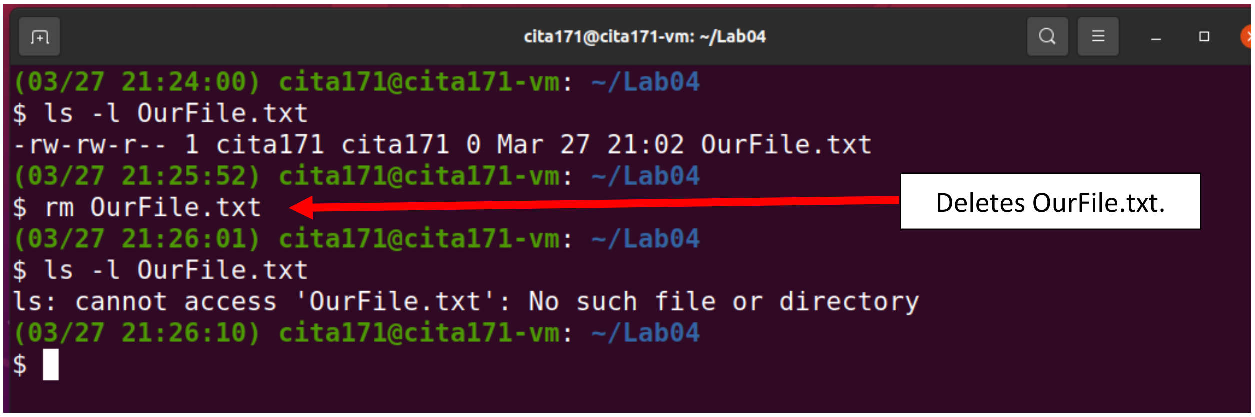
```
(03/27 21:14:18) cita171@cita171-vm: ~/Lab04
$ ls -l
total 4
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 15:05 Archive
-rw-rw-r-- 1 cita171 cita171    0 Mar 27 21:02 OurFile.txt
-rw-rw-r-- 1 cita171 cita171    0 Mar 27 14:27 YourFile.txt
(03/27 21:14:21) cita171@cita171-vm: ~/Lab04
$ cp -r Archive Backups
(03/27 21:14:37) cita171@cita171-vm: ~/Lab04
$ ls -l
total 8
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 15:05 Archive
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 21:14 Backups
-rw-rw-r-- 1 cita171 cita171    0 Mar 27 21:02 OurFile.txt
-rw-rw-r-- 1 cita171 cita171    0 Mar 27 14:27 YourFile.txt
(03/27 21:14:40) cita171@cita171-vm: ~/Lab04
$
```

Copies Archives as Backups. (Note the directory paths are relative.)

Figure 12. The cp Command to Copy a Directory

17 Deleting a File

The **rm** command is used to delete a file. In this example, *OurFile.txt* is deleted. See Figure 13. Note that the ls command fails with an error message indicating that the file no longer exists are the deletion.



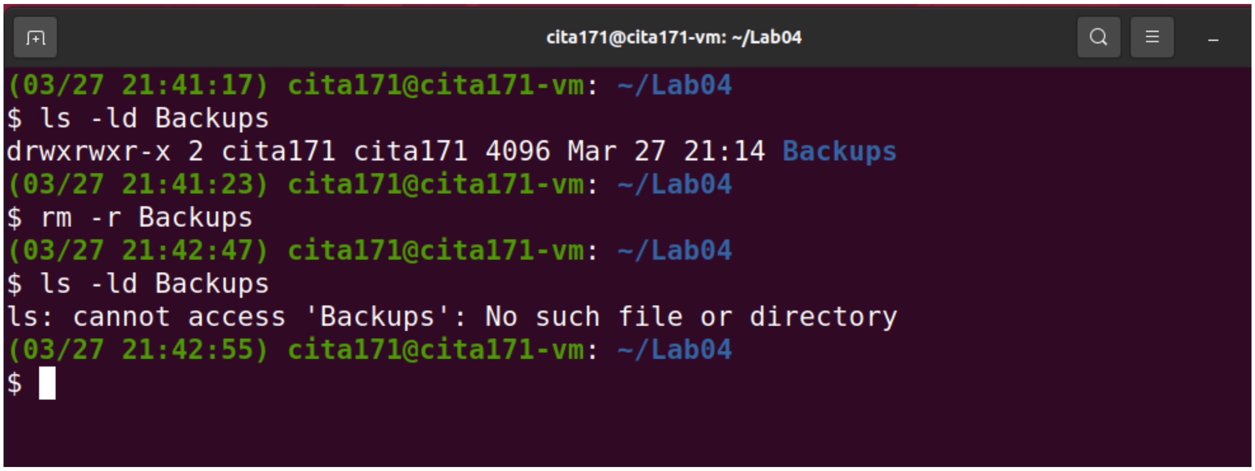
```
(03/27 21:24:00) cita171@cita171-vm: ~/Lab04
$ ls -l OurFile.txt
-rw-rw-r-- 1 cita171 cita171 0 Mar 27 21:02 OurFile.txt
(03/27 21:25:52) cita171@cita171-vm: ~/Lab04
$ rm OurFile.txt
(03/27 21:26:01) cita171@cita171-vm: ~/Lab04
$ ls -l OurFile.txt
ls: cannot access 'OurFile.txt': No such file or directory
(03/27 21:26:10) cita171@cita171-vm: ~/Lab04
$ 
```

Deletes OurFile.txt.

Figure 13. The rm Command to Delete a File

18 DELETING A DIRECTORY

The rm command is also used to delete a directory. However, the -r (recursive delete) option must be added. In this example, the Backups directory is deleted. See Figure 14. Note that the ls command fails with an error message indicating that the directory no longer exists after the deletion.



```
(03/27 21:41:17) cita171@cita171-vm: ~/Lab04
$ ls -ld Backups
drwxrwxr-x 2 cita171 cita171 4096 Mar 27 21:14 Backups
(03/27 21:41:23) cita171@cita171-vm: ~/Lab04
$ rm -r Backups
(03/27 21:42:47) cita171@cita171-vm: ~/Lab04
$ ls -ld Backups
ls: cannot access 'Backups': No such file or directory
(03/27 21:42:55) cita171@cita171-vm: ~/Lab04
$ 
```

Figure 14. The rm Command to Delete a Directory