



VIETNAM – KOREA UNIVERSITY OF  
INFORMATION AND COMMUNICATION TECHNOLOGY  
DEPARTMENT OF COMPUTER SCIENCE

# **REPORT**

## **PROGRAMMING SYSTEM**

**TOPIC: WRITE OR MODIFY FUNCTIONS RELATED  
TO FILE MANAGEMENT - PROGRAMMING THE  
EXECUTION PROCESS**

Group member (Group 7)	: Phạm Thanh Trường Phan Văn Bằng Phan Văn Lai Hồ Thắng Trình
Instructor	: TS. Đặng Quang Hiến

**Da Nang, December 23, 2022**

VIETNAM – KOREA UNIVERSITY OF  
INFORMATION AND COMMUNICATION TECHNOLOGY  
**DEPARTMENT OF COMPUTER SCIENCE**

# **REPORT**

## **PROGRAMMING SYSTEM**

**TOPIC: WRITE OR MODIFY FUNCTIONS RELATED  
TO FILE MANAGEMENT. PROGRAMMING THE  
EXECUTION PROCESS**

Group member (Group 7)	: Phạm Thanh Trường Phan Văn Bằng Phan Văn Lai Hồ Thắng Trình
Instructor	: TS. Đặng Quang Hiến

**Da Nang, December 23, 2022**

## **OPENING**

System programming is an essential and important foundation in any computer's application development, and always evolving to accommodate changes in the computer hardware. System programs provide an environment where programs can be developed and executed. In the simplest sense, system programs also provide a bridge between the user interface and system calls.

The main aim of system programming is to design of system software and to provide basic for judgment in the design of software. Understanding how to properly program systems is a critical skill for developers. Therefore, our group has decided to choose the topic: "Write or modify functions related to file management - programming the execution process". To understand programming principles and techniques for system programming, basic components, and functions of operating systems.

## THANKS

We would like to thank the Department of Computer Science, Vietnam - Korea University of Information and Communication Technology for creating conditions for us to prepare this Report. We would like to express our sincere thanks to our teacher Mr. Dang Quang Hien has enthusiastically guided and advised us during the implementation of the project. In the process of implementing the thesis, we have learned a lot of useful knowledge and valuable experience as the foundation for the process of working and researching later. I would also like to thank the teachers in the Department who have dedicatedly taught and equipped me with important knowledge in the past time. Sincere thanks to all of you, especially the members of the group, who united, supported, and helped us during our time at the school and during the implementation of this thesis. Although we have tried to complete the application within the scope and ability, it is inevitable that there will be shortcomings. We look forward to receiving your understanding and suggestions. We will try harder to improve and develop our topic. Our team sincerely thanks you!

**COMMENT**

(Of Instructor)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Da Nang .....

Instructor

TS. Đặng Quang Hiện

# TABLE OF CONTENTS

<b>OPENING .....</b>	<b>1</b>
<b>LIST OF PICTURES .....</b>	<b>10</b>
<b>PART I: INTRODUCE .....</b>	<b>11</b>
<b>1.1. Topic name.....</b>	<b>11</b>
<b>1.2. Group Member.....</b>	<b>11</b>
<b>PART II: THEORY .....</b>	<b>12</b>
<b>2.1. Linux File Ownership .....</b>	<b>12</b>
<i>2.1.1. User .....</i>	<i>12</i>
<i>2.1.2. Group .....</i>	<i>12</i>
<i>2.1.3. Other .....</i>	<i>12</i>
<b>2.2. Linux File Permissions.....</b>	<b>12</b>
<i>2.2.1. Changing file/directory permissions in Linux .....</i>	<i>14</i>
<i>2.2.2. Changing Ownership and Group in Linux.....</i>	<i>16</i>
<b>PART III: RESULT .....</b>	<b>18</b>
<b>3.1. Write or modify functions related to file management .....</b>	<b>18</b>
<i>3.1.1. Modify function “cat” in Linux.....</i>	<i>18</i>
<i>3.1.2. Write function “ls-group7” same as “ls” command.....</i>	<i>19</i>
<i>3.1.3. Write function “cd-group7” same as “cd” command.....</i>	<i>20</i>
<i>3.1.4. Address of Linux system commands .....</i>	<i>21</i>
<b>3.2. Change Linux File Permissions .....</b>	<b>22</b>
<b>PART IV: CONCLUSION.....</b>	<b>23</b>
<b>5.1. Result .....</b>	<b>23</b>
<b>5.2. Limit .....</b>	<b>23</b>
<b>5.3. Development .....</b>	<b>23</b>
<b>REFERENCES.....</b>	<b>24</b>

## LIST OF PICTURES

<i>Figure 1. Linux file permission.....</i>	<i>13</i>
<i>Figure 2. Change file permission .....</i>	<i>15</i>
<i>Figure 3. Change file permission .....</i>	<i>16</i>
<i>Figure 4. Changing Ownership.....</i>	<i>17</i>
<i>Figure 5. Changing Group.....</i>	<i>17</i>
<i>Figure 6. Modify function “cat” in Linux .....</i>	<i>18</i>
<i>Figure 7. Source code Modify function “cat” in Linux.....</i>	<i>18</i>
<i>Figure 8. Write function “ls-group7” same as “ls” command.....</i>	<i>19</i>
<i>Figure 9. Source code Write function “ls-group7” same as “ls” command .....</i>	<i>19</i>
<i>Figure 10. Write function “cd-group7” same as “cd” command.....</i>	<i>20</i>
<i>Figure 11. Source code Write function “cd-group7” same as “cd” command.....</i>	<i>20</i>
<i>Figure 12. Address of Linux system commands .....</i>	<i>21</i>
<i>Figure 13. Address of Linux system commands in /bin .....</i>	<i>21</i>
<i>Figure 14. Change access permission.....</i>	<i>22</i>
<i>Figure 15. Changing Ownership and Group in Linux .....</i>	<i>22</i>

## **PART I: INTRODUCE**

### **1.1. Topic name**

File management - Write or modify functions related to file management. In addition, programming the execution process (reading files, changing file contents, changing permissions, ...)

### **1.2. Group Member**

- Phạm Thanh Trường – 20IT461
- Phan Văn Bằng - 20IT490
- Phan Văn Lai – 20IT1028
- Hồ Thắng Trình – 20IT511



## PART II: THEORY

Linux is a clone of UNIX, the multi-user operating system which can be accessed by many users simultaneously. Linux can also be used in mainframes and servers without any modifications. But this raises security concerns as an unsolicited or malign user can corrupt, change or remove crucial data. For effective security, Linux divides authorization into 2 levels:

- Ownership
- Permission

### 2.1. Linux File Ownership

Every file and directory on your Unix/Linux system is assigned 3 types of owner, given below.

#### *2.1.1. User*

A user is the owner of the file. By default, the person who created a file becomes its owner. Hence, a user is also sometimes called an owner.

#### *2.1.2. Group*

A user- group can contain multiple users. All users belonging to a group will have the same Linux group permissions access to the file. Suppose you have a project where a number of people require access to a file. Instead of manually assigning permissions to each user, you could add all users to a group, and assign group permission to file such that only this group members and no one else can read or modify the files.

#### *2.1.3. Other*

Any other user who has access to a file. This person has neither created the file, nor he belongs to a user group who could own the file. Practically, it means everybody else. Hence, when you set the permission for others, it is also referred as set permissions for the world.

### 2.2. Linux File Permissions

Every file and directory in your UNIX/Linux system has following 3 permissions defined for all the 3 owners discussed above.

- **Read:** This permission give you the authority to open and read a file. Read permission on a directory gives you the ability to lists its content.
- **Write:** The write permission gives you the authority to modify the contents of a file. The write permission on a directory gives you the authority to add, remove and rename files stored in the directory. Consider a scenario where you have to write permission on file but do not have write permission on the directory where the file is stored. You will be able to modify the file contents. But you will not be able to rename, move or remove the file from the directory.

- **Execute:** In Windows, an executable program usually has an extension “.exe” and which you can easily run. In Unix/Linux, you cannot run a program unless the execute permission is set. If the execute permission is not set, you might still be able to see/modify the program code(provided read & write permissions are set), but not run it.

## Owners assigned Permission On Every File and Directory

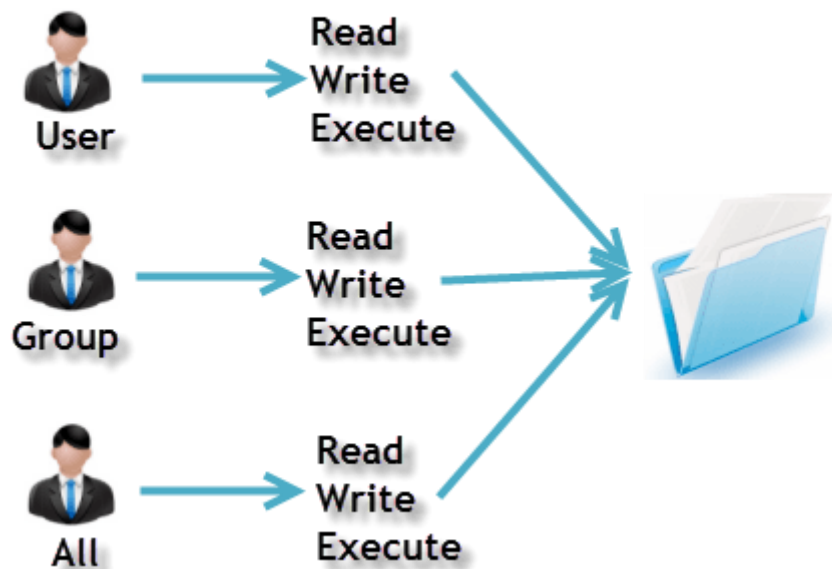


Figure 1. Linux file permission

Syntax show file permissions in Linux:

```
ls -l
```

Here, we have highlighted ‘-rw-rw-r-’ and this weird looking code is the one that tells us about the Unix permissions given to the owner, user group and the world.

Here, the first ‘-’ implies that we have selected a file.p>

```
-rw-rw-r--
```



indicates  
file

Else, if it were a directory, **d** would have been shown.

**d** represents directory

```
drwxr-xr-x 2 ubuntu ubuntu 80 Sep 6 07:27 Desktop
```

**r** = read permission  
**w** = write permission  
**x** = execute permission  
**-** = no permission

### 2.2.1. Changing file/directory permissions in Linux

Say you do not want your colleague to see your personal images. This can be achieved by changing file permissions.

We can use the '**chmod**' command which stands for 'change mode'. Using the command, we can set permissions (read, write, execute) on a file/directory for the owner, group and the world

Syntax:

```
chmod premissions filename
```

There are 2 ways to use the command:

- Absolute mode
- Symbolic mode

#### 2.2.1.1. Absolute (Numeric) Mode in Linux

In this mode, file permissions are not represented as characters but a three-digit octal number. The table below gives numbers for all for permissions types.

Number	Permission Type	Symbol
0	No Permission	—
1	Execute	-X
2	Write	-W-
3	Execute + Write	-WX
4	Read	r-
5	Read + Execute	r-X
6	Read +Write	rw-
7	Read + Write +Execute	rwX

Let's see the chmod permissions command in action:

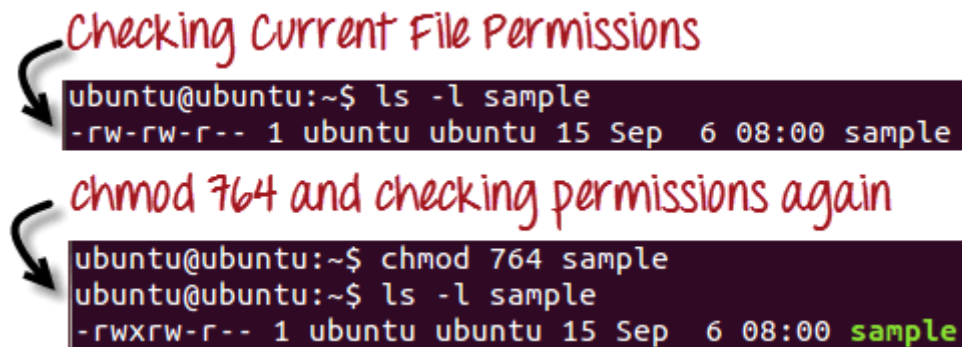
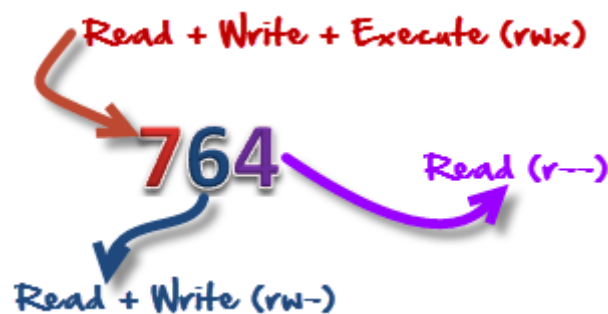


Figure 2. Change file permission

In the above-given terminal window, we have changed the permissions of the file 'sample' to '764'



'764' absolute code says the following:

- Owner can read, write and execute
- Usergroup can read and write
- World can only read

**This is shown as '-rwxrw-r--'**

This is how you can change user permissions in Linux on file by assigning an absolute number.

### 2.2.1.2. Symbolic Mode in Linux

In the Absolute mode, you change permissions for all 3 owners. In the symbolic mode, you can modify permissions of a specific owner. It makes use of mathematical symbols to modify the Unix file permissions.

Operator	Description
+	Adds a permission to a file or directory
-	Removes the permission
=	Sets the permission and overrides the permissions set earlier.

The various owners are represented as –

User Denotations	
u	user/owner
g	group
o	other
a	all

We will not be using permissions in numbers like 755 but characters like rwx. Let's look into an example:

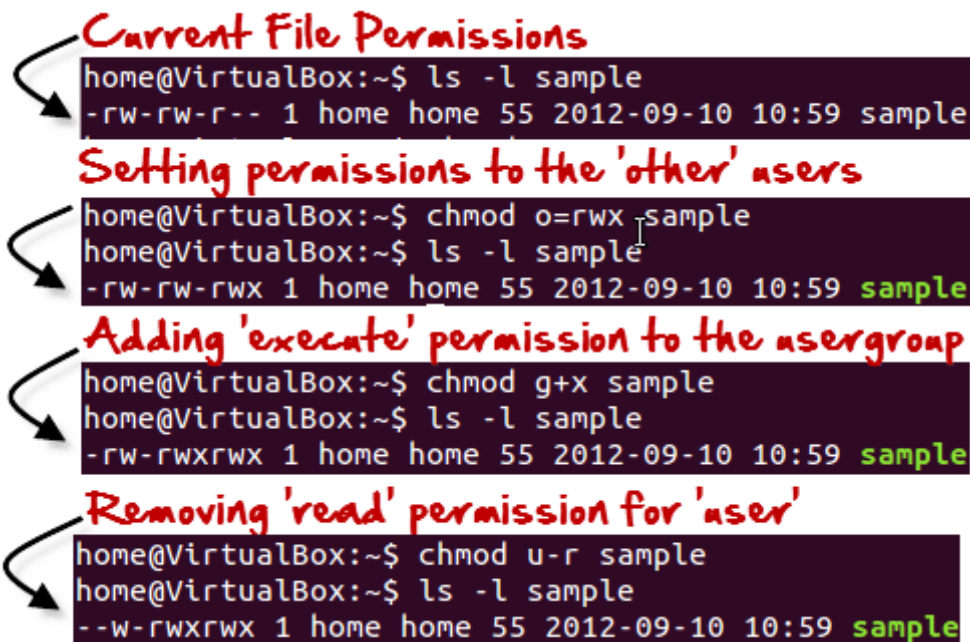


Figure 3. Change file permission

### 2.2.2. Changing Ownership and Group in Linux

For changing the ownership of a file/directory, you can use the following command:

```
chown user filename
```

In case you want to change the user as well as group for a file or directory use the command:

```
chown user:group filename
```

Let's see this in action:

check the current file ownership using `ls -l`

```
-rw-rw-r-- 1 root n10 18 2012-09-16 18:17 sample.txt
```

change the file owner to n100. You will need sudo

```
n10@N100:~$ sudo chown n100 sample.txt
```

ownership changed to n100

```
-rw-rw-r-- 1 n100 n10 18 2012-09-16 18:17 sample.txt
```

changing user and group to root 'chown user:group file'

```
n10@N100:~$ sudo chown root:root sample.txt
```

user and Group ownership changed to root

```
-rw-rw-r-- 1 root root 18 2012-09-16 18:17 sample.txt
```

Figure 4. Changing Ownership

In case you want to change group-owner only, use the command:  
`chgrp group_name filename`

'**chgrp**' stands for change group.

check the current file ownership using `ls -dl`

```
guru99@VirtualBox:~$ ls -dl test1  
-rwxrwxrwx 1 root cdrom 0 Oct 6 11:27 test1
```

change the file owner to root. You will need sudo

```
guru99@VirtualBox:~$ sudo chgrp root test1
```

Group ownership changed to root

```
guru99@VirtualBox:~$ ls -dl test1  
-rwxrwxrwx 1 root root 0 Oct 6 11:27 test1
```

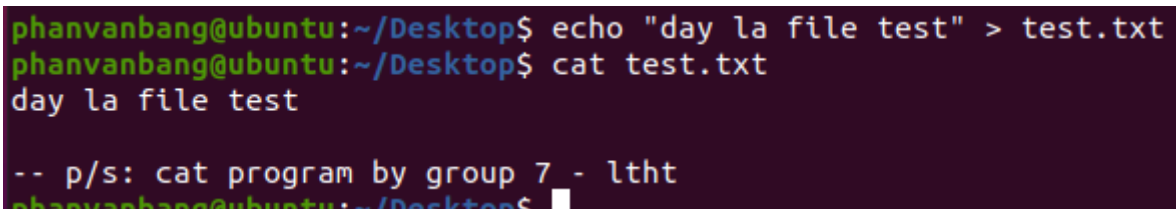
Figure 5. Changing Group

## PART III: RESULT

### 3.1. Write or modify functions related to file management

#### 3.1.1. Modify function "cat" in Linux

Our team changed the "cat" command in the Linux system file. After calling the "cat" command, the terminal still displays the contents of the file, but the bottom also writes the words "-- p/s: cat program by group 7 - ltht"



```
phanvanbang@ubuntu:~/Desktop$ echo "day la file test" > test.txt
phanvanbang@ubuntu:~/Desktop$ cat test.txt
day la file test

-- p/s: cat program by group 7 - ltht
phanvanbang@ubuntu:~/Desktop$
```

Figure 6. Modify function "cat" in Linux

Source code:

```
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>

#define BUFFER_SIZE 50

int main(int argc, char **argv) {
    int file;
    char buffer[BUFFER_SIZE];
    int read_size;

    if (argc < 2) {
        fprintf(stderr, "Error: usage: ./cat filename\n");
        return (-1);
    }

    file = open(argv[1], O_RDONLY);

    if (file == -1) {
        fprintf(stderr, "Error: %s: file not found\n", argv[1]);
        return (-1);
    }

    while ((read_size = read(file, buffer, BUFFER_SIZE)) > 0)
        write(1, &buffer, read_size);

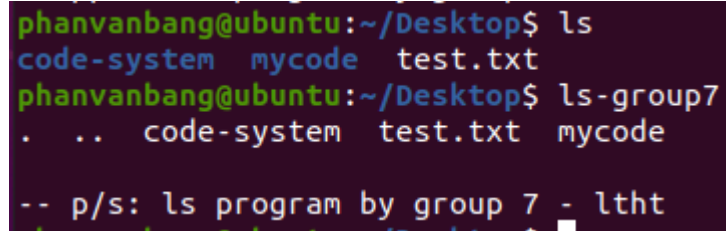
    fprintf(stderr, "\n-- p/s: cat program by group 7 - ltht\n");

    close(file);
    return (0);
}
```

Figure 7. Source code Modify function "cat" in Linux

### 3.1.2. Write function “ls-group7” same as “ls” command

Our team wrote the command "ls-group7" which has the same effect as the command "ls" in the Linux operating system. After calling the command "ls-group7", the terminal still displays files and folders, but the bottom also says "-- p/s: ls program by group 7 - ltht"



```
phanvanbang@ubuntu:~/Desktop$ ls
code-system  mycode  test.txt
phanvanbang@ubuntu:~/Desktop$ ls-group7
.  ..  code-system  test.txt  mycode

-- p/s: ls program by group 7 - ltht
```

Figure 8. Write function “ls-group7” same as “ls” command

Source code:

```
#include <sys/types.h>
#include <dirent.h>
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char **argv) {
    DIR *dp;
    struct dirent *dirp;

    if (argc != 2) {
        fprintf(stderr, "usage: %s dir_name\n", argv[0]);
        exit(1);
    }

    if ((dp = opendir(argv[1])) == NULL) {
        fprintf(stderr, "can't open '%s'\n", argv[1]);
        exit(1);
    }

    while ((dirp = readdir(dp)) != NULL)
        printf("%s ", dirp->d_name);

    printf("\n\n-- p/s: ls program by group 7 - ltht\n");

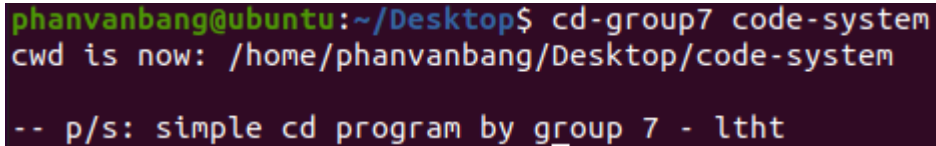
    closedir(dp);
    return(0);
}
```

Figure 9. Source code Write function “ls-group7” same as “ls” command



### 3.1.3. Write function "cd-group7" same as "cd" command

Our team wrote the command "cd-group7" which has the same effect as the command "cd" in the Linux operating system. After calling the command "cd-group7", the terminal displays the path after moving in, but the bottom also says "-- p/s: cd program by group 7 - ltht"

A terminal window with a dark background. The prompt is 'phanvanbang@ubuntu:~/Desktop\$'. The command 'cd-group7 code-system' has been entered. The output shows the current working directory: 'cwd is now: /home/phanvanbang/Desktop/code-system'. Below this, a message is printed: '-- p/s: simple cd program by group 7 - ltht'.

```
phanvanbang@ubuntu:~/Desktop$ cd-group7 code-system
cwd is now: /home/phanvanbang/Desktop/code-system
-- p/s: simple cd program by group 7 - ltht
```

Figure 10. Write function "cd-group7" same as "cd" command

Source code:

```
#include <sys/param.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main(int argc, char **argv) {
    char buf[MAXPATHLEN];

    if (argc != 2) {
        fprintf(stderr, "%s: need a directory!\n", argv[0]);
        return(EXIT_FAILURE);
    }

    if (chdir(argv[1]) == -1) {
        fprintf(stderr, "%s: unable to chdir to %s\n", argv[0], argv[1]);
        return(EXIT_FAILURE);
    }

    printf("cwd is now: %s\n", getcwd(buf, MAXPATHLEN));

    fprintf(stderr, "\n-- p/s: simple cd program by group 7 - ltht\n");

    exit(EXIT_SUCCESS);
}
```

Figure 11. Source code Write function "cd-group7" same as "cd" command

### 3.1.4. Address of Linux system commands

The address of system commands in Linux is at /bin

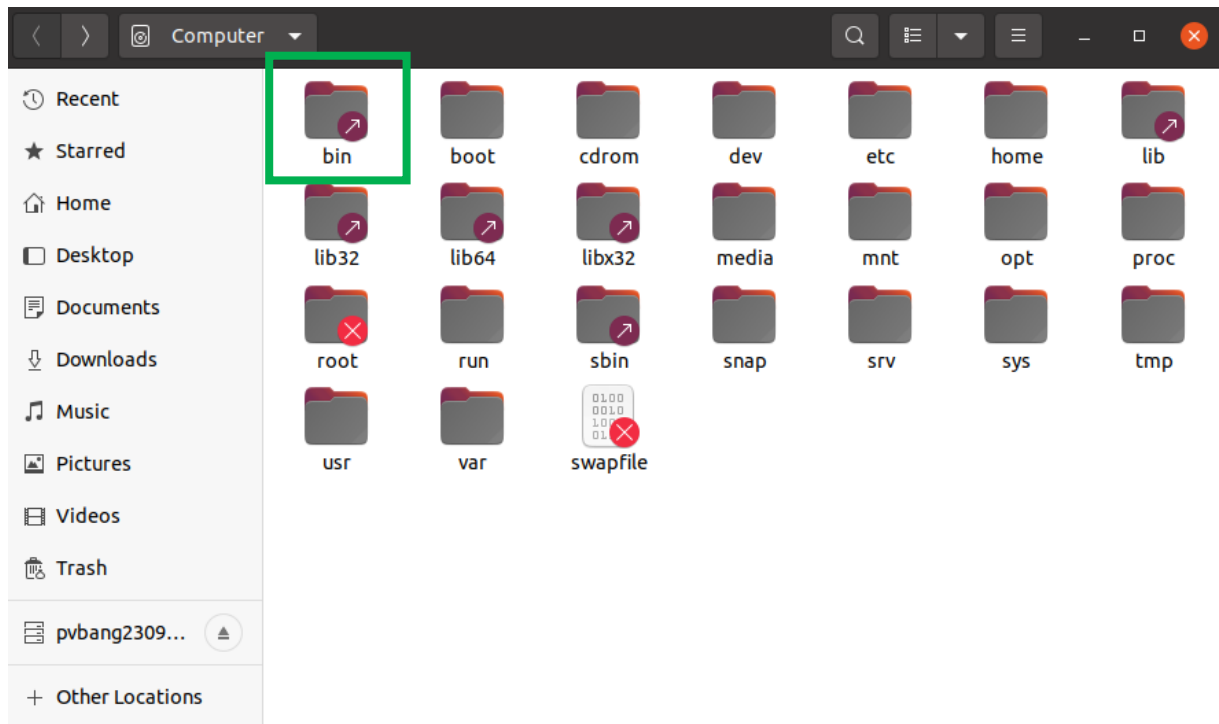


Figure 12. Address of Linux system commands

/bin folder

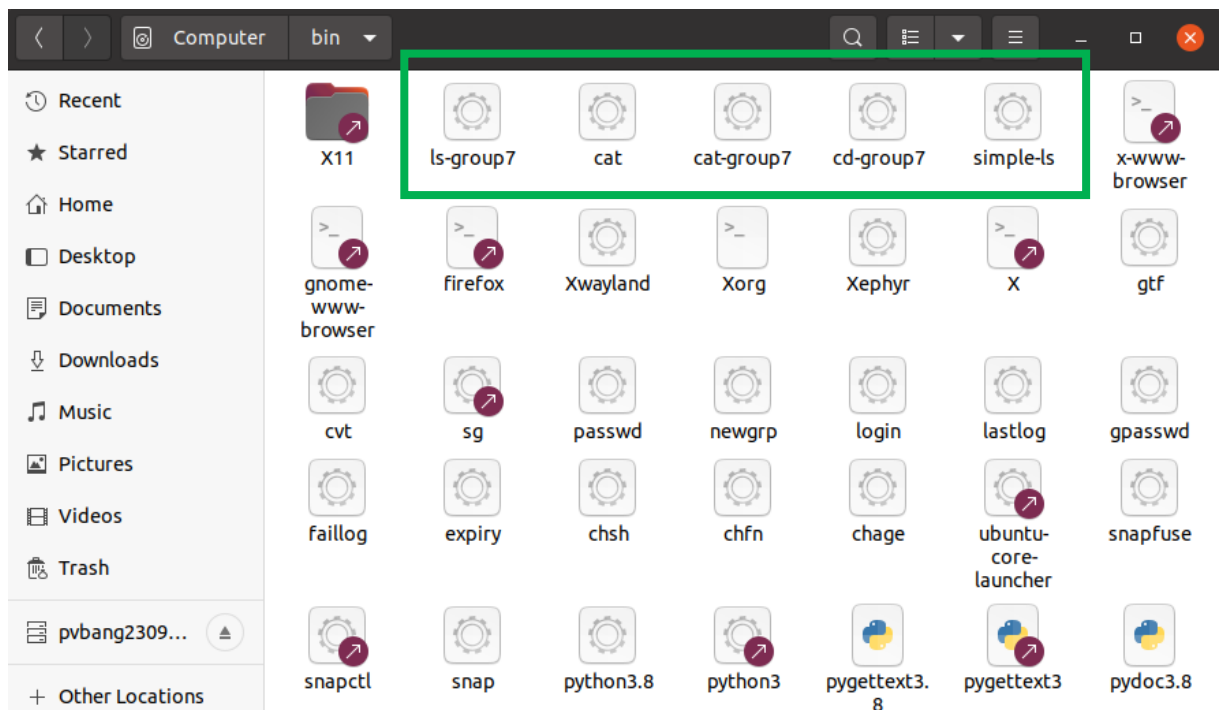


Figure 13. Address of Linux system commands in /bin

### 3.2. Change Linux File Permissions

Change access permission of the file “changing-permissions.txt” to 000. The chmod 000 is not read, not write, not execute.

```
phanvanbang@ubuntu:~/Desktop/code-system/changing-permissions$ ls
changing-permissions.txt
phanvanbang@ubuntu:~/Desktop/code-system/changing-permissions$ ls -l
total 0
-rw-rw-r-- 1 phanvanbang phanvanbang 0 Dec 22 09:10 changing-permissions.txt
phanvanbang@ubuntu:~/Desktop/code-system/changing-permissions$ chmod 000 changing-permissions.txt
phanvanbang@ubuntu:~/Desktop/code-system/changing-permissions$ ls -l
total 0
----- 1 phanvanbang phanvanbang 0 Dec 22 09:10 changing-permissions.txt
```

*Figure 14. Change access permission*

Changing Ownership and Group in Linux:

```
phanvanbang@ubuntu:~/Desktop/code-system/changing-permissions$ ls -l
total 0
----- 1 phanvanbang phanvanbang 0 Dec 22 09:10 changing-permissions.txt
phanvanbang@ubuntu:~/Desktop/code-system/changing-permissions$ sudo chown bang changing-permissions.txt
phanvanbang@ubuntu:~/Desktop/code-system/changing-permissions$ ls -l
total 0
----- 1 bang phanvanbang 0 Dec 22 09:10 changing-permissions.txt
phanvanbang@ubuntu:~/Desktop/code-system/changing-permissions$ sudo chown root:root changing-permissions.txt
phanvanbang@ubuntu:~/Desktop/code-system/changing-permissions$ ls -l
total 0
----- 1 root root 0 Dec 22 09:10 changing-permissions.txt
```

*Figure 15. Changing Ownership and Group in Linux*

## **PART IV: CONCLUSION**

### **5.1. Result**

Understand programming principles and techniques for system programming, basic components, and functions of operating systems.

Write, compile, debug and execute C programs for system programming and properly use system interfaces provided by UNIX (or UNIX-like Operating System).

Organize and work effectively in groups, practice the ability to use English in presenting and reading documents.

### **5.2. Limit**

Have not set up and build efficient networked multi-process applications.

### **5.3. Development**

Set up and build efficient networked multi-process applications yet.

Learn more about programming principles and techniques for system programming, basic components, and functions of operating systems.

Develop more features for existing and new programs.

## REFERENCES

- [1] Advanced Programming in the UNIX® Environment, W. Richard Stevens Stephen A. Rago, Safari Books, 2000
- [2] The Practice of Programming, Brian W. Kernighan and Rob Pike, O'REILLY, 2014
- [3] The\_C\_Programming\_Language, Edition\_Ritchie\_Kernighan, Springer, 2013
- [4] <https://www.geeksforgeeks.org/c-program-to-find-and-replace-a-word-in-a-file-by-another-given-word/>
- [5] <https://stevens.netmeister.org/765-APUE/>
- [6] <https://www.redhat.com/sysadmin/customizing-linux-filesystem-commands>
- [7] <https://cheatography.com/davechild/cheat-sheets/linux-command-line/>