

National Textile University

**Department of Computer Science**

Subject: Operating System

Submitted to: Sir Nasir Mahmood

Submitted by: Eisha Muzaffar

Reg. number: 23-NTU-CS-1147

Lab no: lab3 hometask

semester:5th

**Operating Systems – COC 3071L**

# SE 5th A – Fall 2025 Part 1: File and Directory Operations

1. Create the following directory structure in your home directory:

Lab

\_

3

/

├──

do

c

s

/

│

└──

dr

a

fts

/

├──

d

a

t

a

/

│

├──

r

a

w

/

│

└──

pro

c

essed

/

└──

s

c

ripts

/

2

.

Inside

do

c

s

/

:

Create three files:

intro

.

txt

,

notes

.

txt

,

summ

a

ry

.

txt

.

Add at least

**two lines of text**

into each using

e

c

ho

>>

.

Copy

summ

a

ry

.

txt

into the

dr

a

fts

/

folder using

c

p

command.

3

.

Inside

d

a

t

a

/

r

a

w

/

:

Create two files:

r

a

w

1

.

txt

,

r

a

w

2

.

txt

.

Append the

**current date**

into

r

a

w

1

.

txt

using the

d

a

te

command.

Move

r

a

w

2

.

txt

into

pro

c

essed

/

using

mv

. The syntax is:

mv

sour

c

e

destin

a

tion

4

.

Inside

s

c

ripts

/

:

Create a script named

hello

.

sh

with the following content:

e

c

ho

"

H

ello

W

orld

"

pwd

ls

-

lh

Later, you will make it executable (in Part 3).

5

.

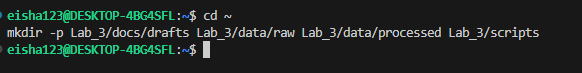
Display the directory structure recursively and take a screenshot:

ls

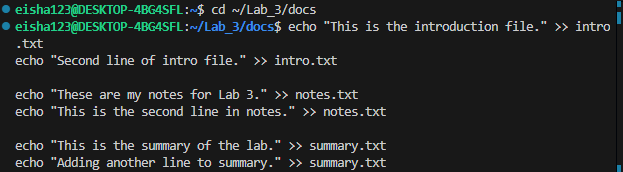
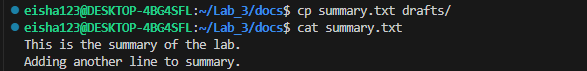
-

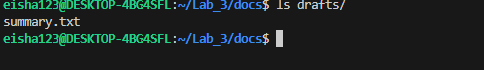
R

1. . Create the following directory structure in your home directory:

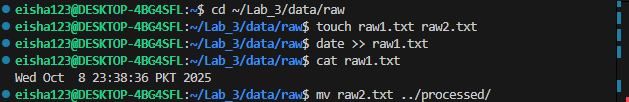


2. Inside docs/:

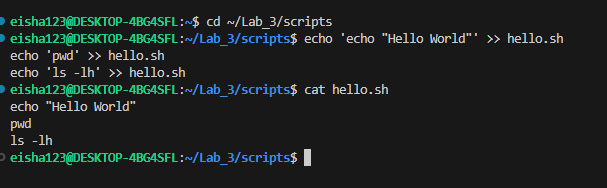
  
2.1 Copy:  




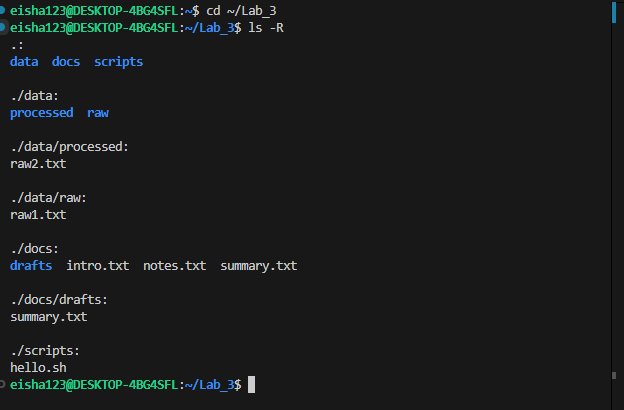
3.Inside data/raw:



4.Inside scripts/ :



5.ls -R:



# Part 2: Practice with Basic Linux Commands

Run the following commands inside Lab\_3/ and note their outputs:

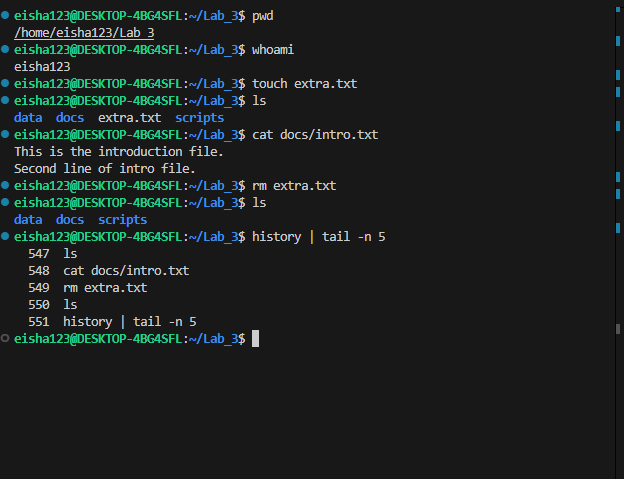
pwd → Show current working directory.

whoami → Display the current logged-in user. touch extra.txt → Create an empty file. cat intro.txt → Display file contents. rm extra.txt → Delete a file.

history | tail -n 5 → Show your last 5 executed commands. clear → Clear the terminal.

Take screenshots of commands and outputs.

**All commands in part 2:**



# Part 3: File Permissions and Ownership

1

.

Change the permissions of

hello

.

sh

so that:

Owner → Read, Write & Execute

Group → Read, Write & Execute

Others → No permissions

Run the script using:

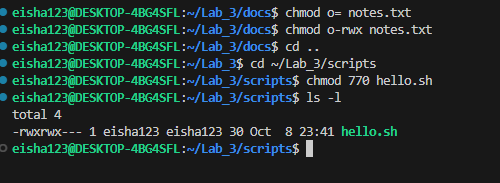
./

hello

.

sh

Take a screenshot of its output.

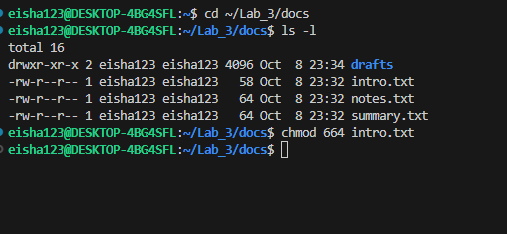


1. Change the permissions of intro.txt using **numeric notation** so that:

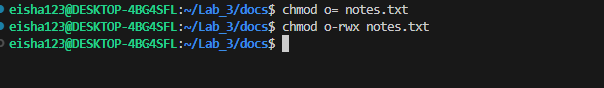
Owner → Read & Write

Group → Read & Write

Others → Read only



1. Change the permissions of notes.txt using **symbolic notation** so that others don't have any permission on it.



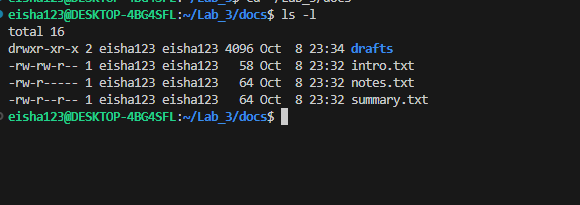
1. Verify all changes with:

ls

-

l

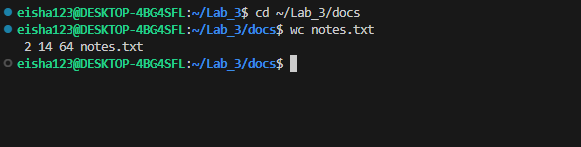
Take a screenshot of the output.



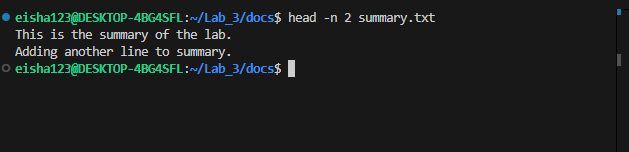
# Part 4: Reading & Searching Files

Inside docs/ :

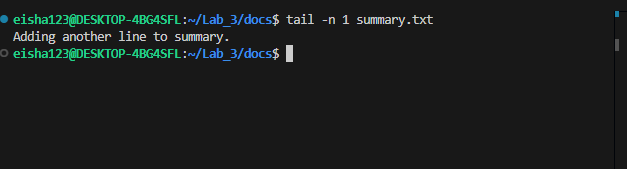
1. Count the number of lines, words, and characters in notes.txt using wc .



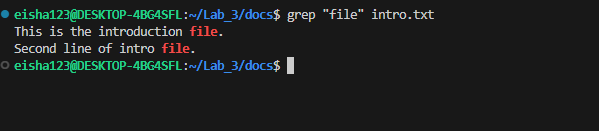
1. Show only the **first 2 lines** of summary.txt using head -n 2 .



1. Show the **last line** of summary.txt using tail -n 1 .



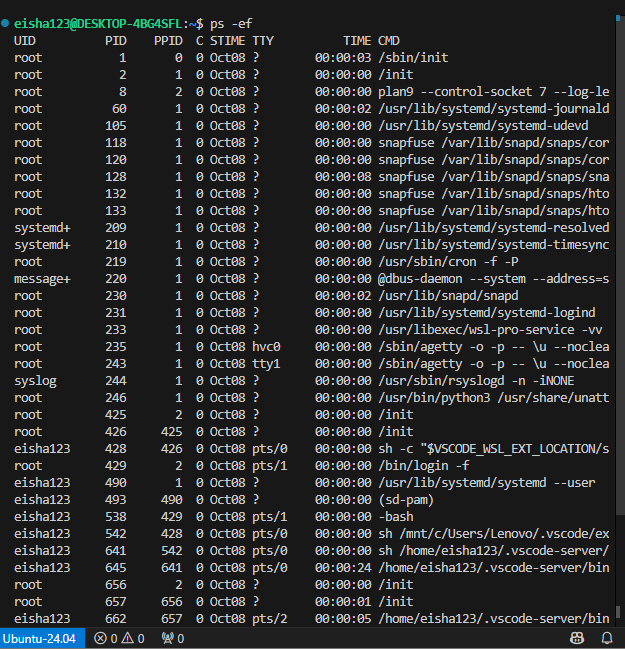
1. Search for a keyword (of your choice) in intro.txt using grep

.

# Part 5: Linux Process Commands

## 1. Exploring Processes

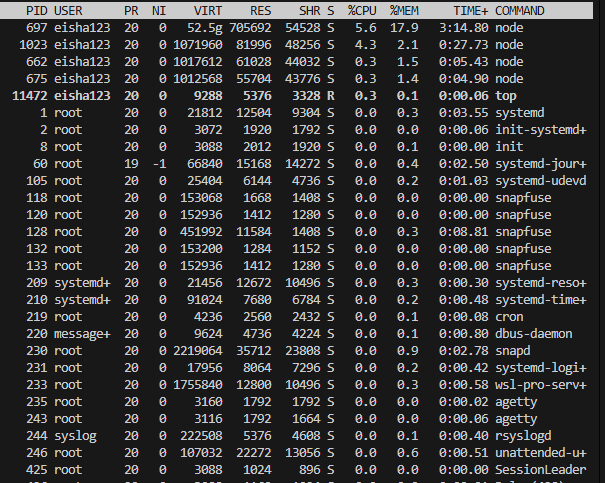
Use ps -ef and identify **3 processes** running on your system. Note their **PID, PPID, and command**.



Run top for 20–30 seconds. Write down:

Which process is consuming the most CPU.

Which process is consuming the most memory.



The process with PID 697 and user eisha123 is using the most memory(17.9) as well as cpu(5.6)

## 2. Practice with Infinite Process

Start:

yes

>

/

dev

/

null

&

Locate its PID using

ps

-

ef

|

grep

yes

.

Kill it using

kill

<

PID

>

and verify using

ps

.

3

.

**Foreground & Background Jobs**

Run

sleep

60

in

**foreground**

and terminate it with

**Ctrl + C**

.

Run

sleep

60

&

in

**background**

, bring it to foreground with

fg

, stop with

**Ctrl + Z**

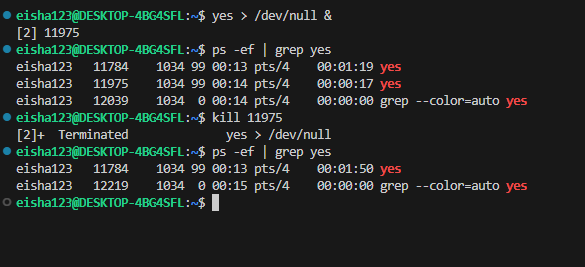
,

then resume in background using

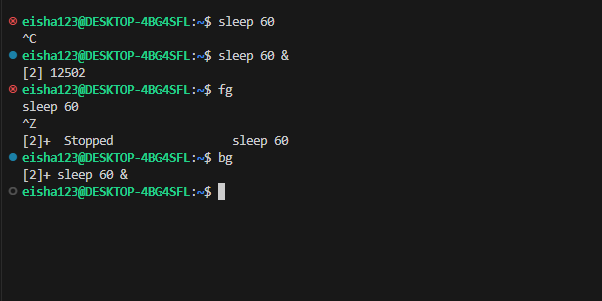
b

g

.

**2.Screenshot for infinite Process:**  


**3.ScreenShots for Foreground and Background jobs:**

****

# Part 6: C Programs on Processes

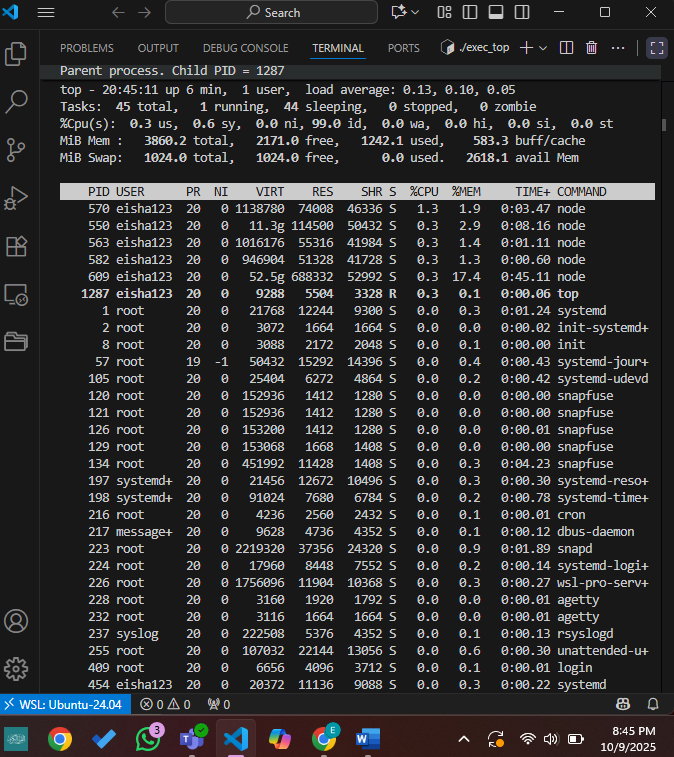
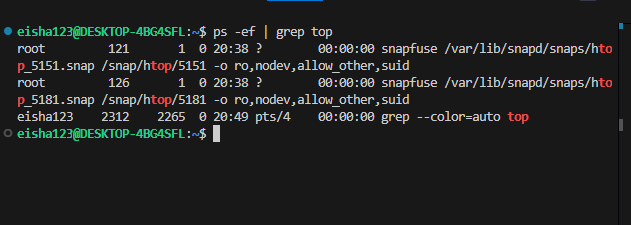
**Program 1 – Exec with top**

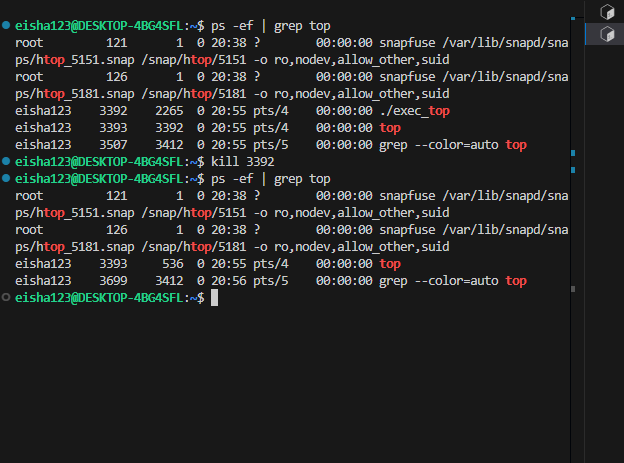
Modify the exec program so that the child runs **top** instead of ls -l .

Run the program.

In another terminal, use ps -ef | grep top (or run top ) to find the child’s PID.

Use the child's process ID to kill it manually.



**Program 2 – Incomplete Program**

#

in

c

lude

<

stdio

.

h

>

#

in

c

lude

<

unistd

.

h

>

#

in

c

lude

<

sys

/

w

a

it

.

h

>

int

m

a

in

(

)

{

pid

\_

t

pid

=

fork

(

)

;

if

(

pid

==

0

)

{

//

TODO

:

R

epl

ac

e

this

c

hild

pro

c

ess

with

the

"

d

a

te

"

c

omm

a

nd

using

exe

c

lp

//

H

int

:

exe

c

lp

(

"

d

a

te

", "

d

a

te

",

NULL

)

;

}

else

{

//

TODO

:

Ma

ke

p

a

rent

w

a

it

for

c

hild

b

efore

printing

"

C

hild

finished

"

}

return

0

;

}

Program:

#include <stdio.h>

#include <unistd.h>

#include <sys/wait.h>

int main() {

pid\_t pid = fork

if (pid == 0) {

// Child process

printf("Child process running 'date'...\n");

execlp("date", "date", NULL); // replace child process with date command

// if execlp fails:

printf("execlp failed\n");

}

else if (pid > 0) {

// Parent process

wait(NULL); // wait for the child process to finish

printf("Child finished.\n");

}

else {

// fork failed

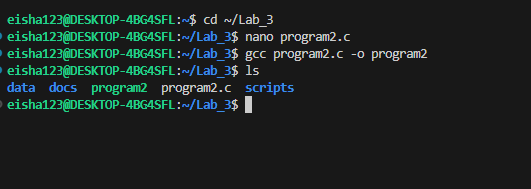
printf("Fork failed!\n");

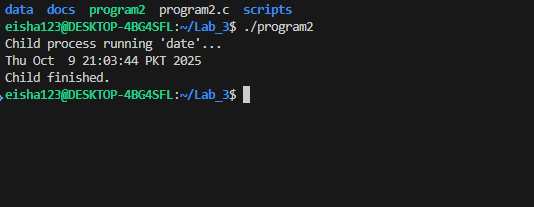
}

return 0;

}

**Task:** Complete the missing parts, run the program, and take a screenshot of the output.





# Submission Guidelines

Submit a **single PDF file** including:

Screenshots of all said commands & outputs.

Modified & completed C program code and outputs.

**Deadline:** 9th October, 2025, 11:59 PM.