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/
├── docs/
│   └── drafts/
├── data/
│   ├── raw/
│   └── processed/
└── scripts/
```

2. Inside `docs/` :
   - Create three files: `intro.txt`, `notes.txt`, `summary.txt` .
   - Add at least **two lines of text** into each using `echo >>` .
   - Copy `summary.txt` into the `drafts/` folder using `cp` command.
3. Inside `data/raw/` :
   - Create two files: `raw1.txt`, `raw2.txt`.
   - Append the **current date** into `raw1.txt` using the `date` command.
   - Move `raw2.txt` into `processed/` using `mv` . The syntax is:

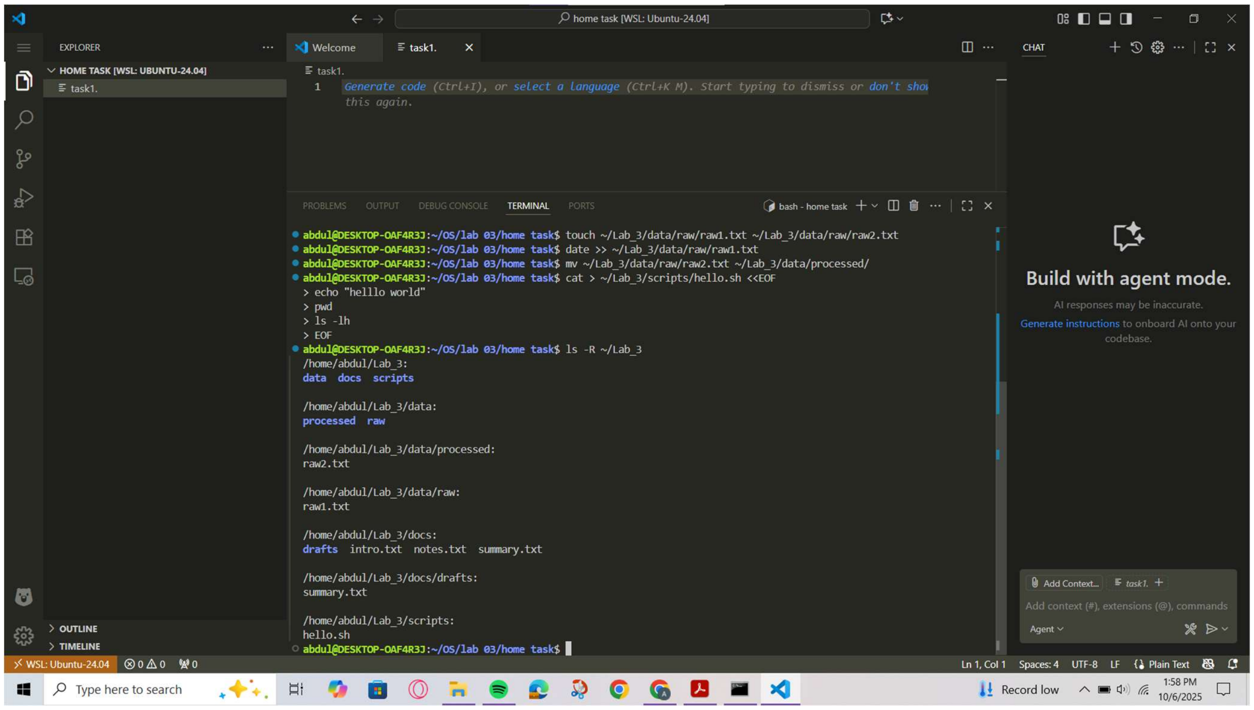
```
mv source destination
```

4. Inside `scripts/` :
   - Create a script named `hello.sh` with the following content:

```
echo "Hello World"
pwd
ls -lh
```

   - Later, you will make it executable (in Part 3).
5. Display the directory structure recursively and take a screenshot:

```
ls -R
```



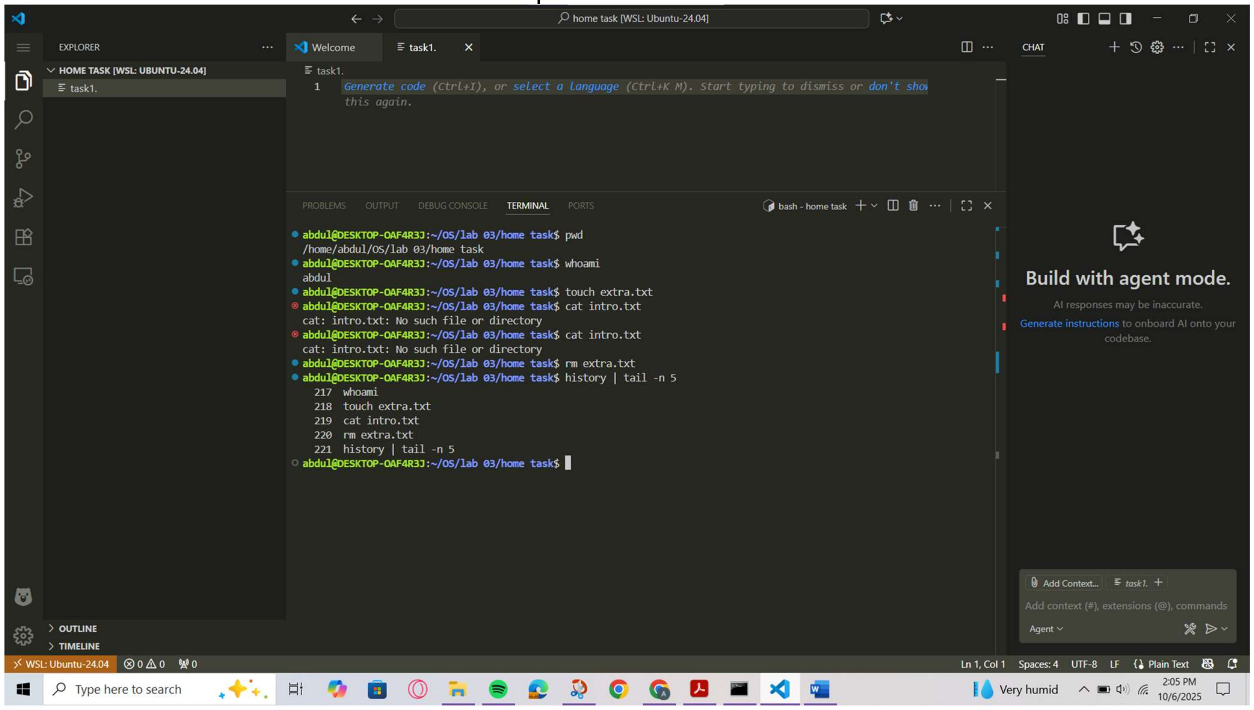
```
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ touch ~/Lab_3/data/raw/raw1.txt ~/Lab_3/data/raw/raw2.txt
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ date >> ~/Lab_3/data/raw/raw1.txt
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ mv ~/Lab_3/data/raw/raw2.txt ~/Lab_3/data/processed/
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ cat > ~/Lab_3/scripts/hello.sh <<EOF
> echo "helllo world"
> pwd
> ls -lh
> EOF
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ ls -R ~/Lab_3
/home/abdul/Lab_3:
data  docs  scripts

/home/abdul/Lab_3/data:
processed  raw

/home/abdul/Lab_3/data/processed:
raw2.txt

/home/abdul/Lab_3/data/raw:
raw1.txt

/home/abdul/Lab_3/docs:
drafts  intro.txt  notes.txt  summary.txt

/home/abdul/Lab_3/docs/drafts:
summary.txt

/home/abdul/Lab_3/scripts:
hello.sh
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$
```

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.

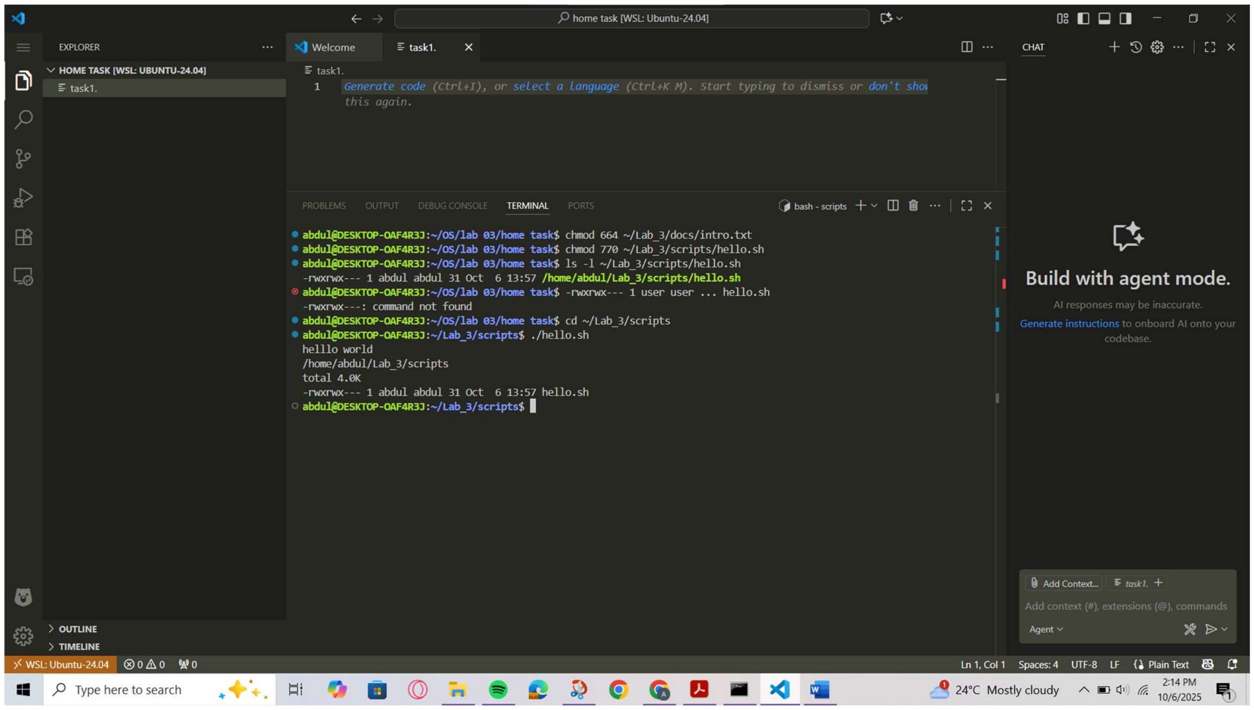
Take screenshots of commands and outputs.


home task [WSL: Ubuntu-24.04]
EXPLORER
HOME TASK [WSL: UBUNTU-24.04]
task1.
Welcome
task1.
Generate code (Ctrl+I), or select a language (Ctrl+K M). Start typing to dismiss or don't show this again.
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
bash - home task
CHAT
Build with agent mode.
AI responses may be inaccurate.
Generate instructions to onboard AI onto your codebase.
Add Context...
Add context (#), extensions (@), commands
Agent
OUTLINE
TIMELINE
WSL: Ubuntu-24.04
Ln 1, Col 1
Spaces: 4
UTF-8
LF
Plain Text
Type here to search
Very humid
2:05 PM
10/6/2025

```
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ pwd
/home/abdul/OS/lab 03/home task
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ whoami
abdul
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ touch extra.txt
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ cat intro.txt
cat: intro.txt: No such file or directory
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ cat intro.txt
cat: intro.txt: No such file or directory
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ rm extra.txt
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ history | tail -n 5
  217  whoami
  218  touch extra.txt
  219  cat intro.txt
  220  rm extra.txt
  221  history | tail -n 5
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ 
```

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:
Take a screenshot of its output.
./hello.sh
2. Change the permissions of intro.txt using numeric notation so that:
Owner → Read & Write
Group → Read & Write
Others → Read only

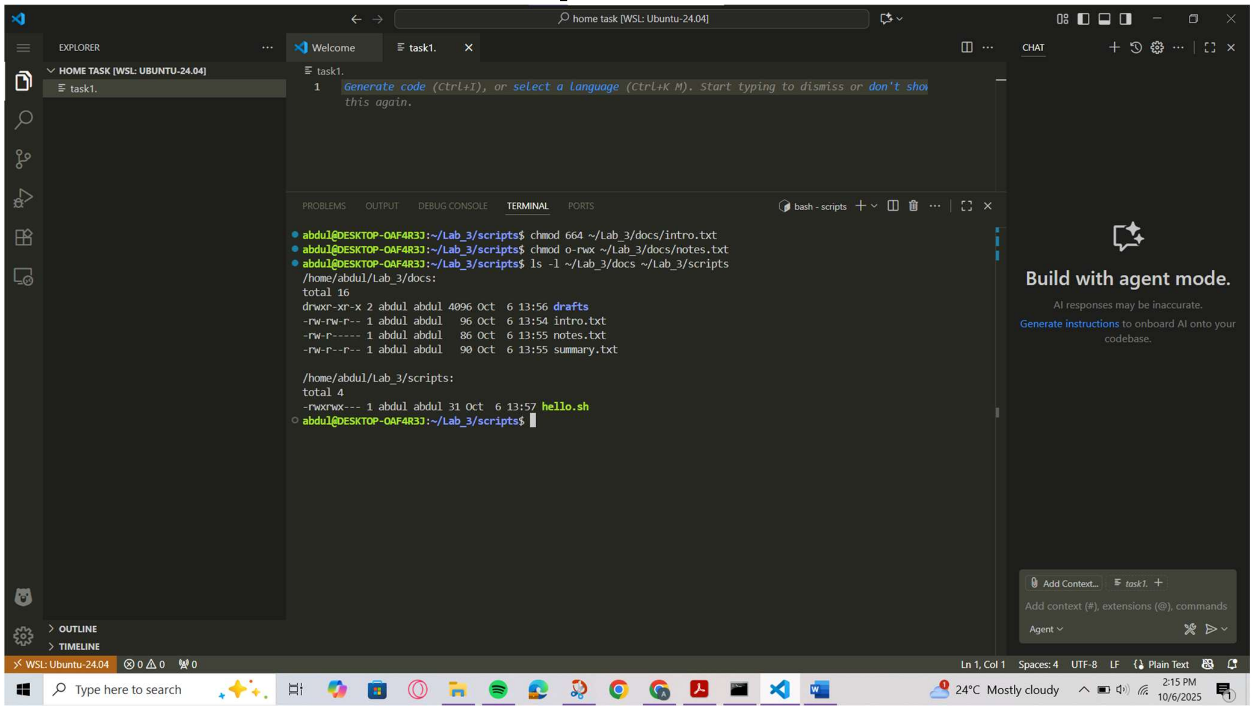


```
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ chmod 664 ~/Lab_3/docs/intro.txt
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ chmod 770 ~/Lab_3/scripts/hello.sh
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ ls -l ~/Lab_3/scripts/hello.sh
-rwxrwx--- 1 abdul abdul 31 Oct  6 13:57 /home/abdul/Lab_3/scripts/hello.sh
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ -rwxrwx--- 1 user user ... hello.sh
-rwxrwx---: command not found
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ cd ~/Lab_3/scripts
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ ./hello.sh
helllo world
/home/abdul/Lab_3/scripts
total 4.0K
-rwxrwx--- 1 abdul abdul 31 Oct  6 13:57 hello.sh
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ 
```

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

4. Verify all changes with:

Take a screenshot of the output.



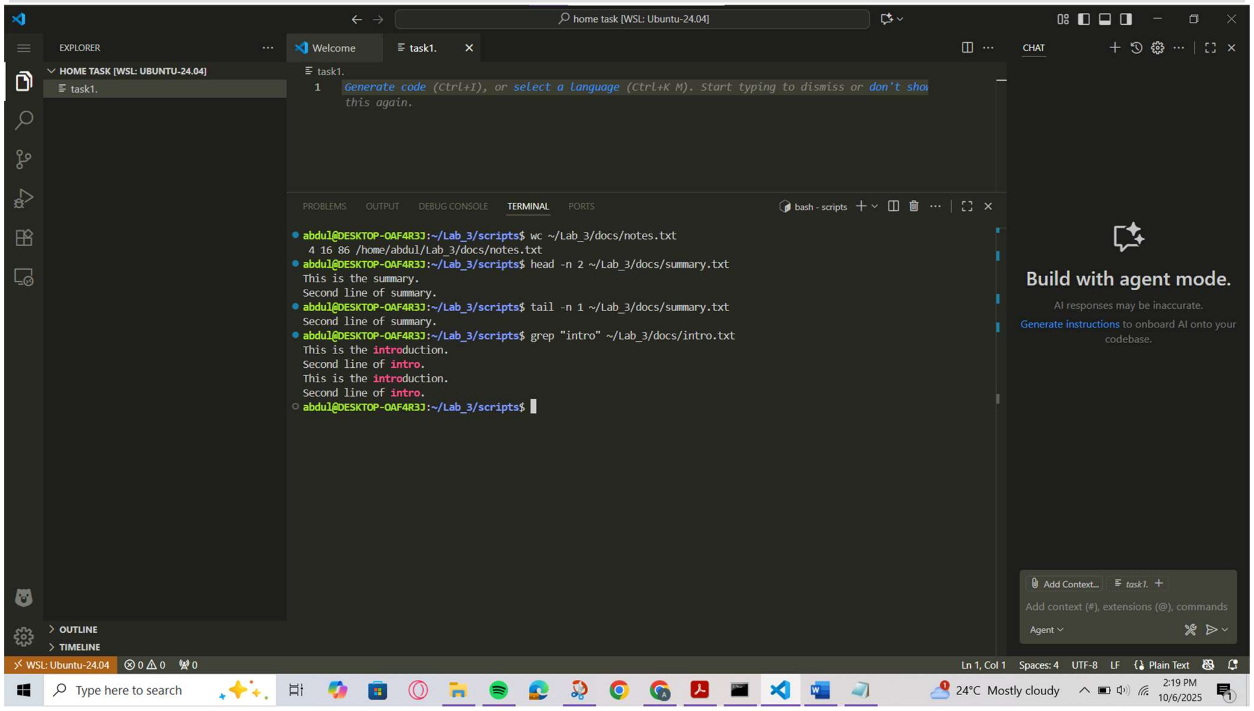
```
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ chmod 664 ~/Lab_3/docs/intro.txt
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ chmod o-rwx ~/Lab_3/docs/notes.txt
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ ls -l ~/Lab_3/docs ~/Lab_3/scripts
/home/abdul/Lab_3/docs:
total 16
drwxr-xr-x 2 abdul abdul 4096 Oct  6 13:56 drafts
-rw-rw-r-- 1 abdul abdul   96 Oct  6 13:54 intro.txt
-rw-r----- 1 abdul abdul   86 Oct  6 13:55 notes.txt
-rw-r--r-- 1 abdul abdul   90 Oct  6 13:55 summary.txt

/home/abdul/Lab_3/scripts:
total 4
-rwxrwx--- 1 abdul abdul 31 Oct  6 13:57 hello.sh
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ 
```

Part 4: Reading & Searching Files

Inside `docs/`:

1. Count the number of lines, words, and characters in `notes.txt` using `wc`.
2. Show only the **first 2 lines** of `summary.txt` using `head -n 2`.
3. Show the **last line** of `summary.txt` using `tail -n 1`.
4. Search for a keyword (of your choice) in `intro.txt` using `grep`.

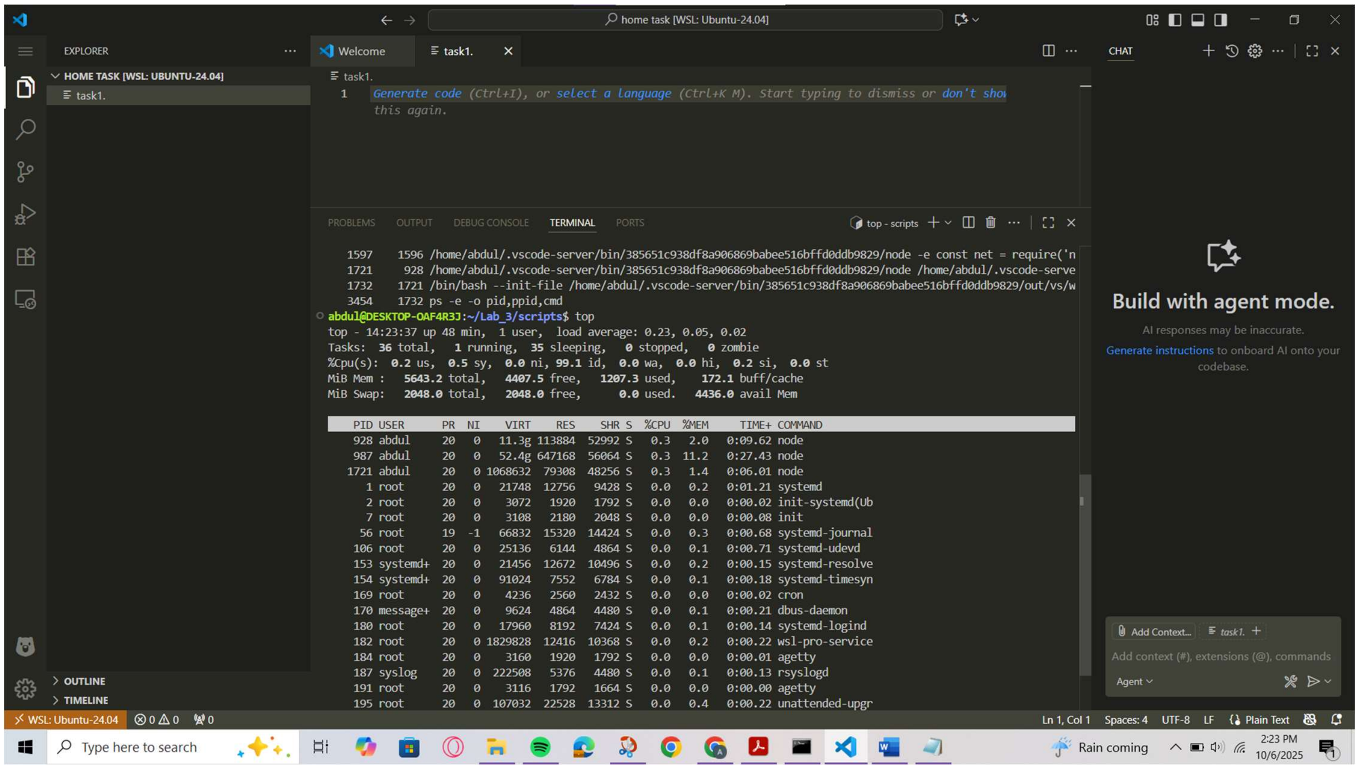
Take screenshots.



```
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ wc ~/Lab_3/docs/notes.txt
 4 16 86 /home/abdul/Lab_3/docs/notes.txt
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ head -n 2 ~/Lab_3/docs/summary.txt
This is the summary.
Second line of summary.
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ tail -n 1 ~/Lab_3/docs/summary.txt
Second line of summary.
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ grep "intro" ~/Lab_3/docs/intro.txt
This is the introduction.
Second line of intro.
This is the introduction.
Second line of intro.
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$
```

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.

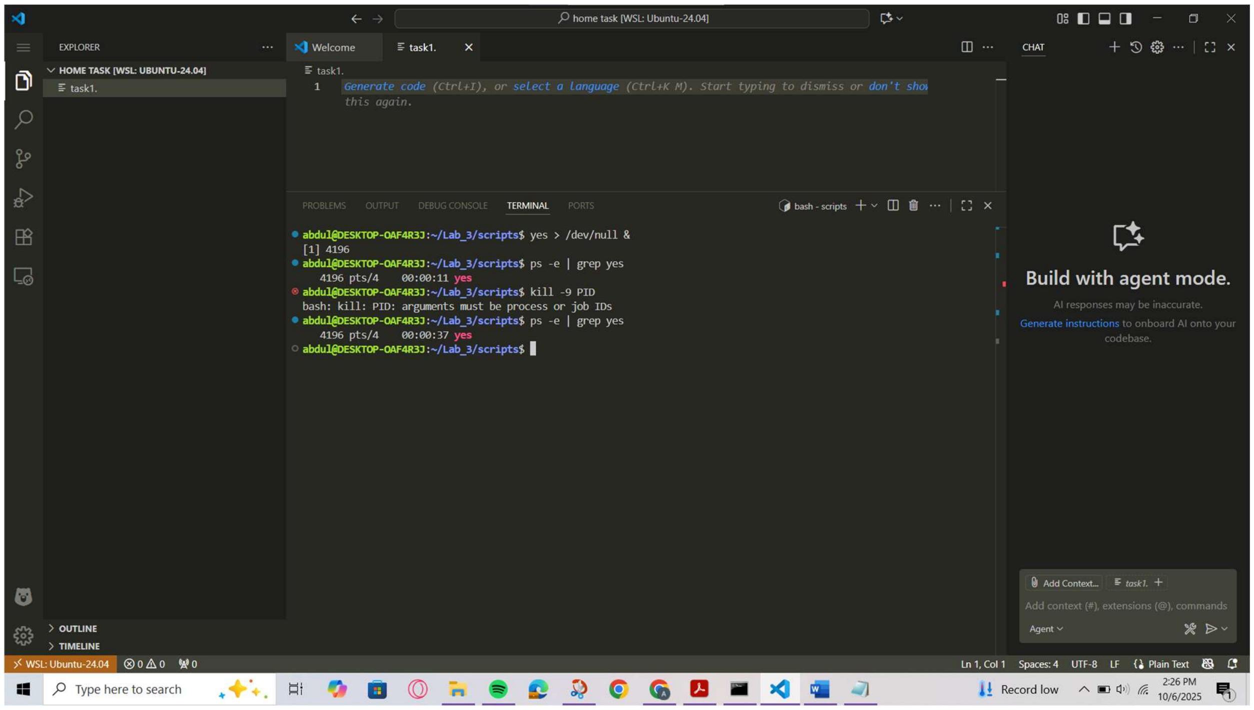


```
   1597   1596 /home/abdul/.vscode-server/bin/385651c938df8a906869babee516bffd0ddb9829/node -e const net = require('n
   1721    928 /home/abdul/.vscode-server/bin/385651c938df8a906869babee516bffd0ddb9829/node /home/abdul/.vscode-serve
   1732   1721 /bin/bash --init-file /home/abdul/.vscode-server/bin/385651c938df8a906869babee516bffd0ddb9829/out/vs/w
   3454   1732 ps -e -o pid,ppid,cmd
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ top
top - 14:23:37 up 48 min,  1 user,  load average: 0.23, 0.05, 0.02
Tasks:  36 total,   1 running,  35 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.2 us,  0.5 sy,  0.0 ni, 99.1 id,  0.0 wa,  0.0 hi,  0.2 si,  0.0 st
MiB Mem :   5643.2 total,   4407.5 free,   1207.3 used,    172.1 buff/cache
MiB Swap:   2048.0 total,   2048.0 free,      0.0 used.   4436.0 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    928 abdul     20   0   11.3g 113884  52992 S   0.3   2.0   0:09.62 node
    987 abdul     20   0   52.4g 647168  56064 S   0.3  11.2   0:27.43 node
   1721 abdul     20   0 1068632  79308  48256 S   0.3   1.4   0:06.01 node
      1 root      20   0   21748  12756   9428 S   0.0   0.2   0:01.21 systemd
      2 root      20   0    3072   1920   1792 S   0.0   0.0   0:00.02 init-systemd(Ub
      7 root      20   0    3108   2180   2048 S   0.0   0.0   0:00.08 init
     56 root      19  -1   66832  15320  14424 S   0.0   0.3   0:00.68 systemd-journal
    106 root      20   0   25136   6144   4864 S   0.0   0.1   0:00.71 systemd-udevd
    153 systemd+  20   0   21456  12672  10496 S   0.0   0.2   0:00.15 systemd-resolve
    154 systemd+  20   0   91024   7552   6784 S   0.0   0.1   0:00.18 systemd-timesyn
    169 root      20   0    4236   2560   2432 S   0.0   0.0   0:00.02 cron
    170 message+  20   0    9624   4864   4480 S   0.0   0.1   0:00.21 dbus-daemon
    180 root      20   0   17960   8192   7424 S   0.0   0.1   0:00.14 systemd-logind
    182 root      20   0 1829828  12416  10368 S   0.0   0.2   0:00.22 wsl-pro-service
    184 root      20   0    3160   1920   1792 S   0.0   0.0   0:00.01 agetty
    187 syslog    20   0  222508   5376   4480 S   0.0   0.1   0:00.13 rsyslogd
    191 root      20   0    3116   1792   1664 S   0.0   0.0   0:00.00 agetty
    195 root      20   0  107032  22528  13312 S   0.0   0.4   0:00.22 unattended-upgr
```

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`.



```
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ yes > /dev/null &
[1] 4196
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ ps -e | grep yes
   4196 pts/4    00:00:11 yes
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ kill -9 PID
bash: kill: PID: arguments must be process or job IDs
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ ps -e | grep yes
   4196 pts/4    00:00:37 yes
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$
```

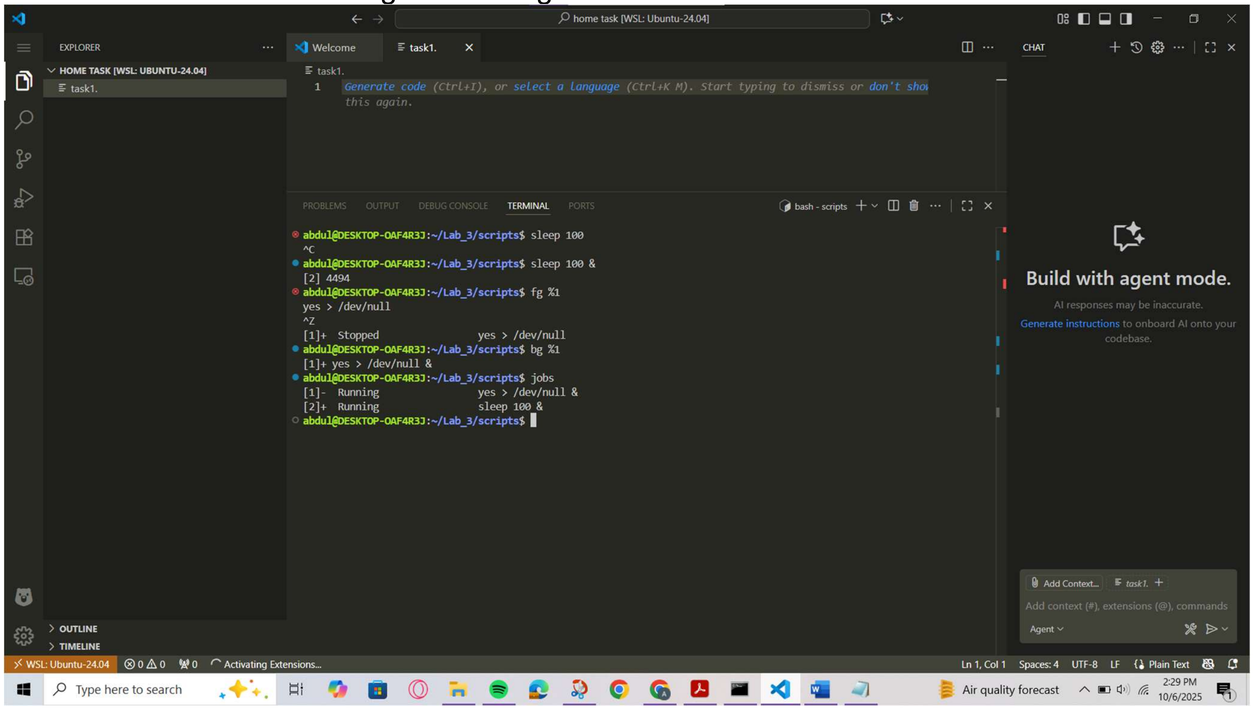
3. **Foreground & Background Jobs**
   - Run `sleep 60` Run in **foreground**
   - `sleep 60 &`

and terminate it with **Ctrl + C**. in **background**, bring it to foregro

und with

`fg`, stop with **Ctrl + Z**,

then resume in background using .



```
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ sleep 100
^C
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ sleep 100 &
[2] 4494
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ fg %1
yes > /dev/null
^Z
[1]+  Stopped                 yes > /dev/null
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ bg %1
[1]+ yes > /dev/null &
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$ jobs
[1]-  Running                 yes > /dev/null &
[2]+  Running                 sleep 100 &
abdul@DESKTOP-OAF4R3J:~/Lab_3/scripts$
```

Part 6: C Programs on Processes

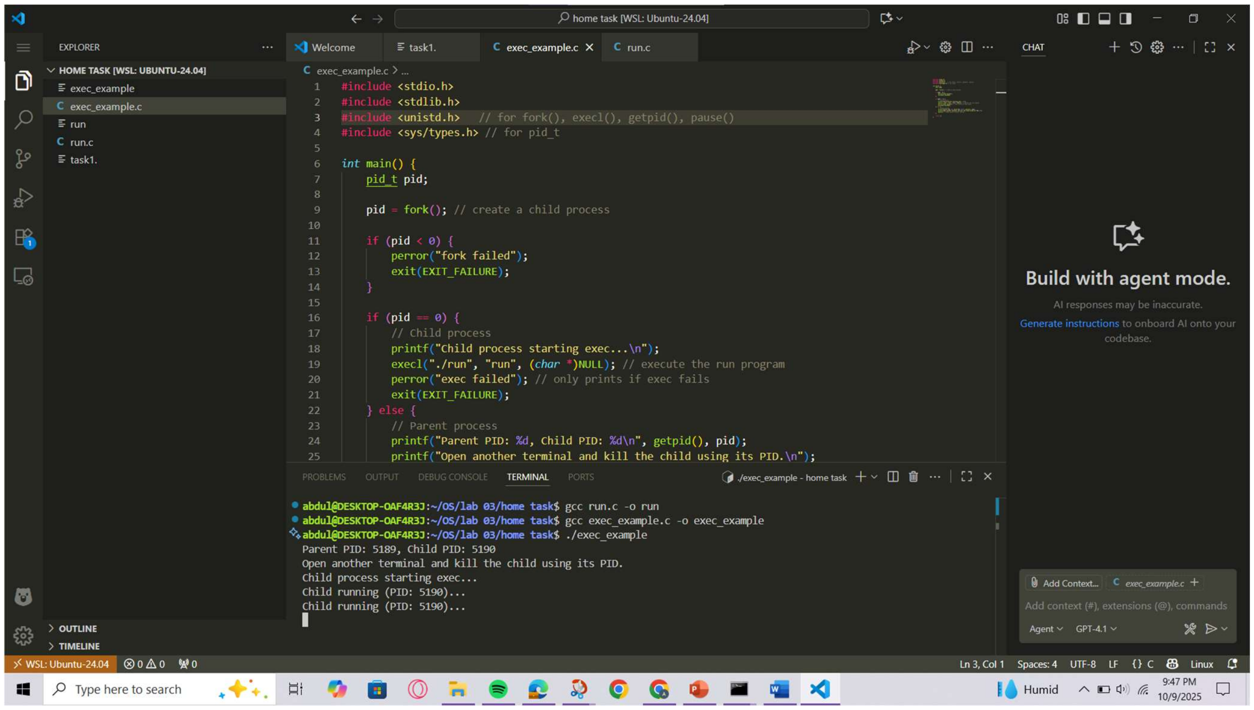
Program 1 – Exec with `top`

- 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

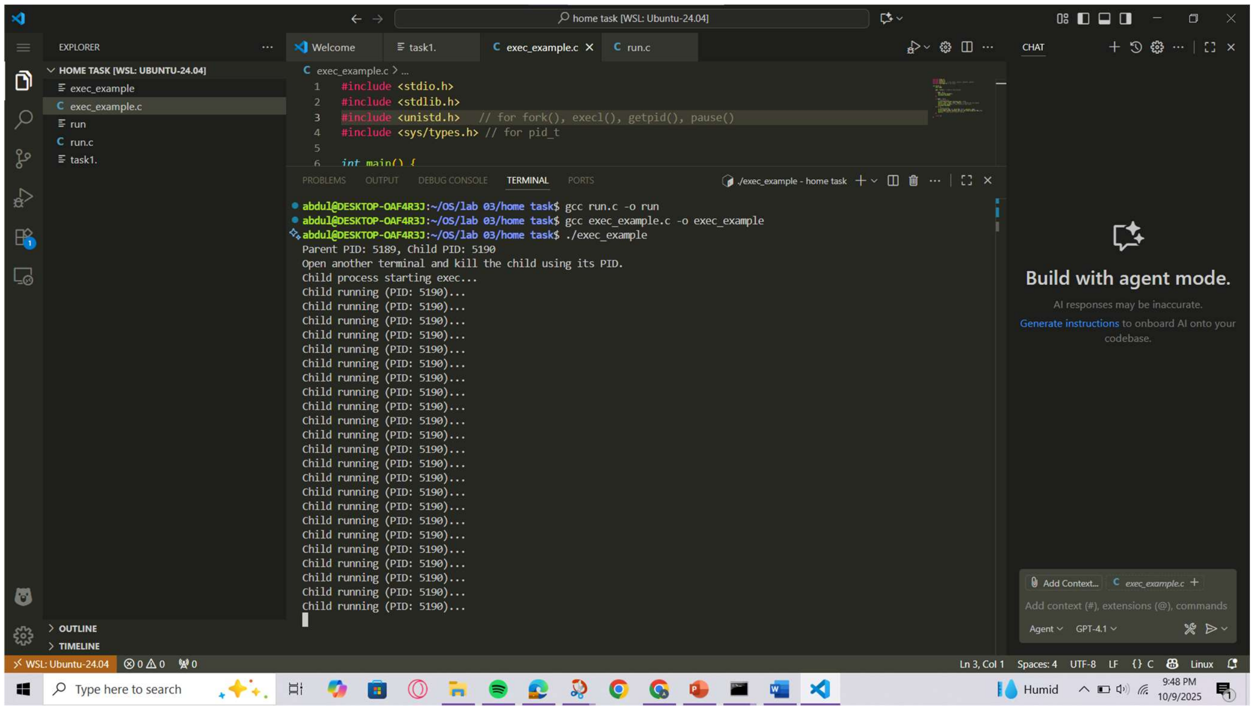
```c
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>

int main() {
    pid_t pid = fork();

    if (pid == 0) {
        // TODO: Replace this child process with the "date" command using
execlp
        // Hint: execlp("date", "date", NULL);
    } else {
        // TODO: Make parent wait for child before printing "Child finished"
    }

    return 0;
}
```

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



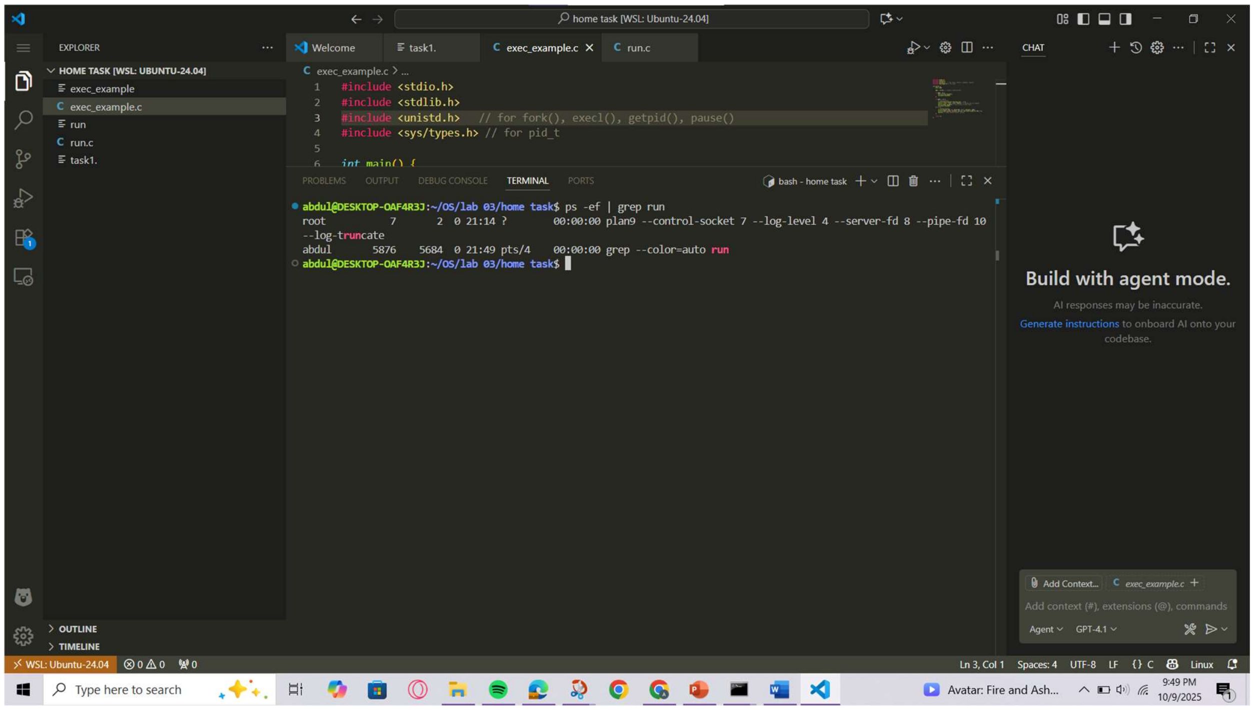
```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>   // for fork(), execl(), getpid(), pause()
#include <sys/types.h> // for pid_t

int main() {
    pid_t pid;

    pid = fork(); // create a child process

    if (pid < 0) {
        perror("fork failed");
        exit(EXIT_FAILURE);
    }

    if (pid == 0) {
        // Child process
        printf("Child process starting exec...\n");
        execl("./run", "run", (char *)NULL); // execute the run program
        perror("exec failed"); // only prints if exec fails
        exit(EXIT_FAILURE);
    } else {
        // Parent process
        printf("Parent PID: %d, Child PID: %d\n", getpid(), pid);
        printf("Open another terminal and kill the child using its PID.\n");
```

```
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ gcc run.c -o run
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ gcc exec_example.c -o exec_example
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ ./exec_example
Parent PID: 5189, Child PID: 5190
Open another terminal and kill the child using its PID.
Child process starting exec...
Child running (PID: 5190)...
Child running (PID: 5190)...
```



```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>   // for fork(), execl(), getpid(), pause()
#include <sys/types.h> // for pid_t

int main() {
```

```
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ gcc run.c -o run
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ gcc exec_example.c -o exec_example
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ ./exec_example
Parent PID: 5189, Child PID: 5190
Open another terminal and kill the child using its PID.
Child process starting exec...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
Child running (PID: 5190)...
```



```
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$ ps -ef | grep run
root           7       2  0 21:14 ?        00:00:00 plan9 --control-socket 7 --log-level 4 --server-fd 8 --pipe-fd 10
--log-truncate
abdul       5876    5684  0 21:49 pts/4    00:00:00 grep --color=auto run
abdul@DESKTOP-OAF4R3J:~/OS/lab 03/home task$
```

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.