/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*

I declare that this lab is my own work in accordance with Seneca Academic Policy.  \*  No part of this assignment has been copied manually or electronically from any other source\*  (including web sites) or distributed to other students.\* \*

Name: Chetan Arora Student ID: 100976240 Date: 12 July 2025

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

1. Why do we check /proc/self/status for memory usage?

Ans:- We check /proc/self/status to monitor the real-time memory usage of the process. This file is part of the Linux procfs and contains detailed information about a process’s resource usage, including VmRSS, which represents the Resident Set Size (i.e., the amount of physical memory currently being used).

In our worker program, reading VmRSS helps us decide if the worker is exceeding the 50MB memory usage threshold, at which point we send a signal (SIGUSR1) to alert the parent.

2. What would happen if kill(parent\_pid, SIGUSR1); fails?

Ans:- If kill(parent\_pid, SIGUSR1); fails, the signal will not be delivered, and the parent will not be notified of the worker's high memory usage.

Possible reasons for failure:

The parent\_pid is incorrect or the parent process no longer exists (ESRCH error).

The worker doesn’t have permission to send signals to the parent (EPERM error).

If it fails:

No log will be recorded.

No action will be taken by the parent, possibly missing a critical event.

It may silently fail unless the error is logged or handled explicitly.

3. How does read() behave differently in blocking vs non-blocking mode?

Ans:-Blocking Mode (default):

* + read() waits (blocks) until data is available.
  + If there is no data to read, the process pauses execution.
  + Good for simple programs where waiting is acceptable.

Non-Blocking Mode (O\_NONBLOCK):

* + read() returns immediately, even if no data is available.
  + If there's no data, it returns -1 and sets errno to EAGAIN or EWOULDBLOCK.
  + Used in event-driven or multiplexed I/O systems (e.g., servers handling multiple clients).

In my assignment, blocking mode is used because we read large binary files from disk sequentially and expect data to always be available.

4. What happens if two workers send SIGUSR1 at the same time?

Ans:- If two workers send SIGUSR1 simultaneously, the Linux kernel queues only one instance of the signal at a time if the signal is not queued (which is the case for standard signals like SIGUSR1).

* The parent process may receive only one signal notification, potentially missing the other.
* If you're using sigaction() with the SA\_SIGINFO flag (which we do), the parent can still identify which worker sent the signal using siginfo\_t.

Possible implications:

* You may lose one of the signals if they arrive too close together.
* For robust designs that require signal queuing, real-time signals (SIGRTMIN+n) should be used instead.

5. Why is signal(SIGUSR1, xx\_signal\_handler); (or sigaction) needed in main()?

Ans:- Without registering a custom signal handler in main():

* The process will use the default action for the signal.
  + For SIGUSR1 and SIGUSR2, the default action is to terminate the process.
* So, if we don’t register xx\_signal\_handler using signal() or sigaction(), the parent will crash as soon as it receives a worker's signal.

By setting a signal handler:

* The parent process can catch the signal, log the event, and continue running.
* It also allows us to identify the sender and perform custom logic (e.g., fcntl logging, printing messages).

6. What is F\_WRLCK and why do we use it?

Ans:- F\_WRLCK stands for Write Lock in the context of the fcntl() file locking mechanism in Unix/Linux systems.

* It is used to lock a region of a file for writing, preventing other processes from writing (or reading, if using mandatory locks) to that region until the lock is released.
* It’s part of the struct flock structure used with fcntl().

Why we use it:

* To ensure that only one process at a time writes to a shared file (e.g., syslog.log).
* This prevents race conditions, where multiple workers or signal handlers might attempt to write log messages simultaneously, potentially corrupting the log.

7. How does file locking prevent data corruption?

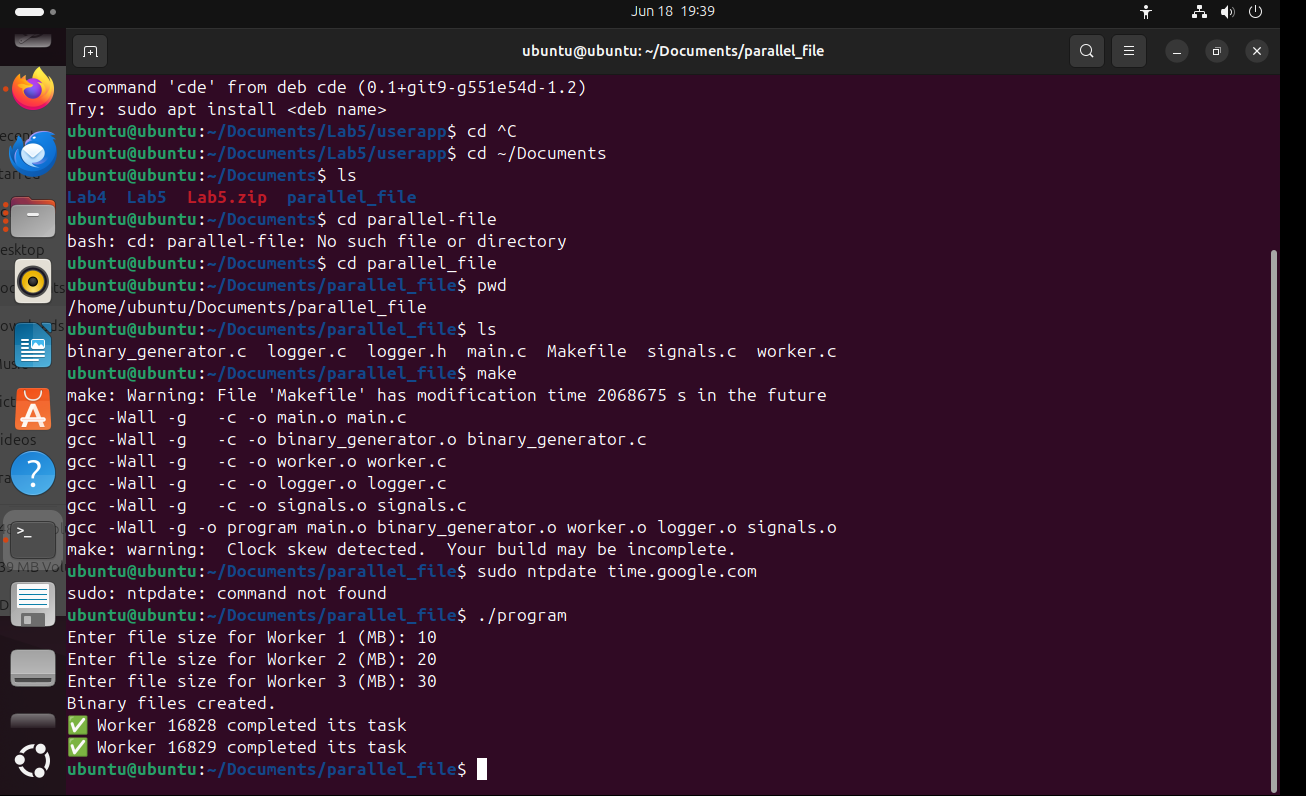
Ans:- File locking (specifically using fcntl() with F\_WRLCK) prevents data corruption by:

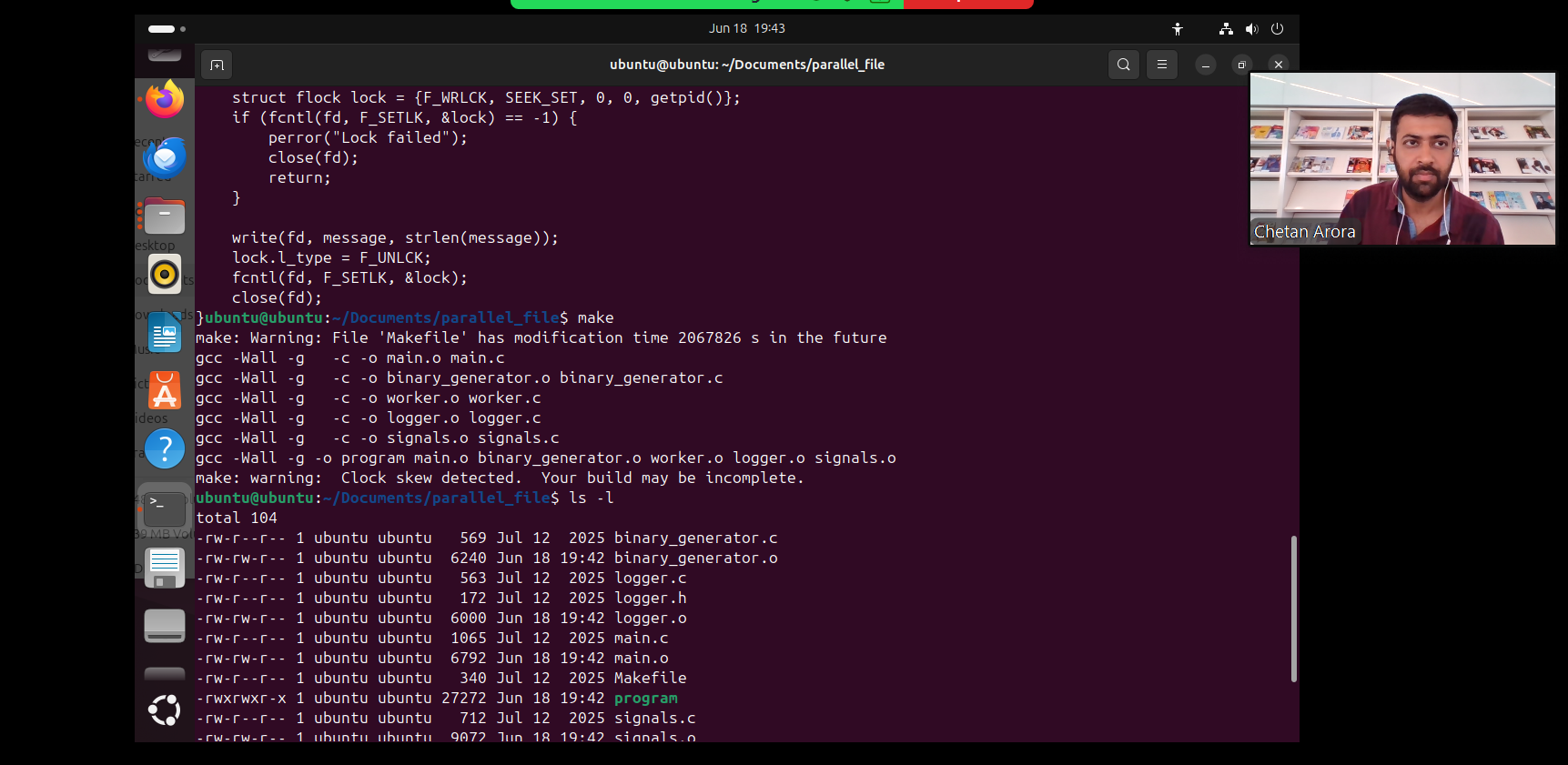
Forcing processes to wait for the lock to be released before accessing the file.

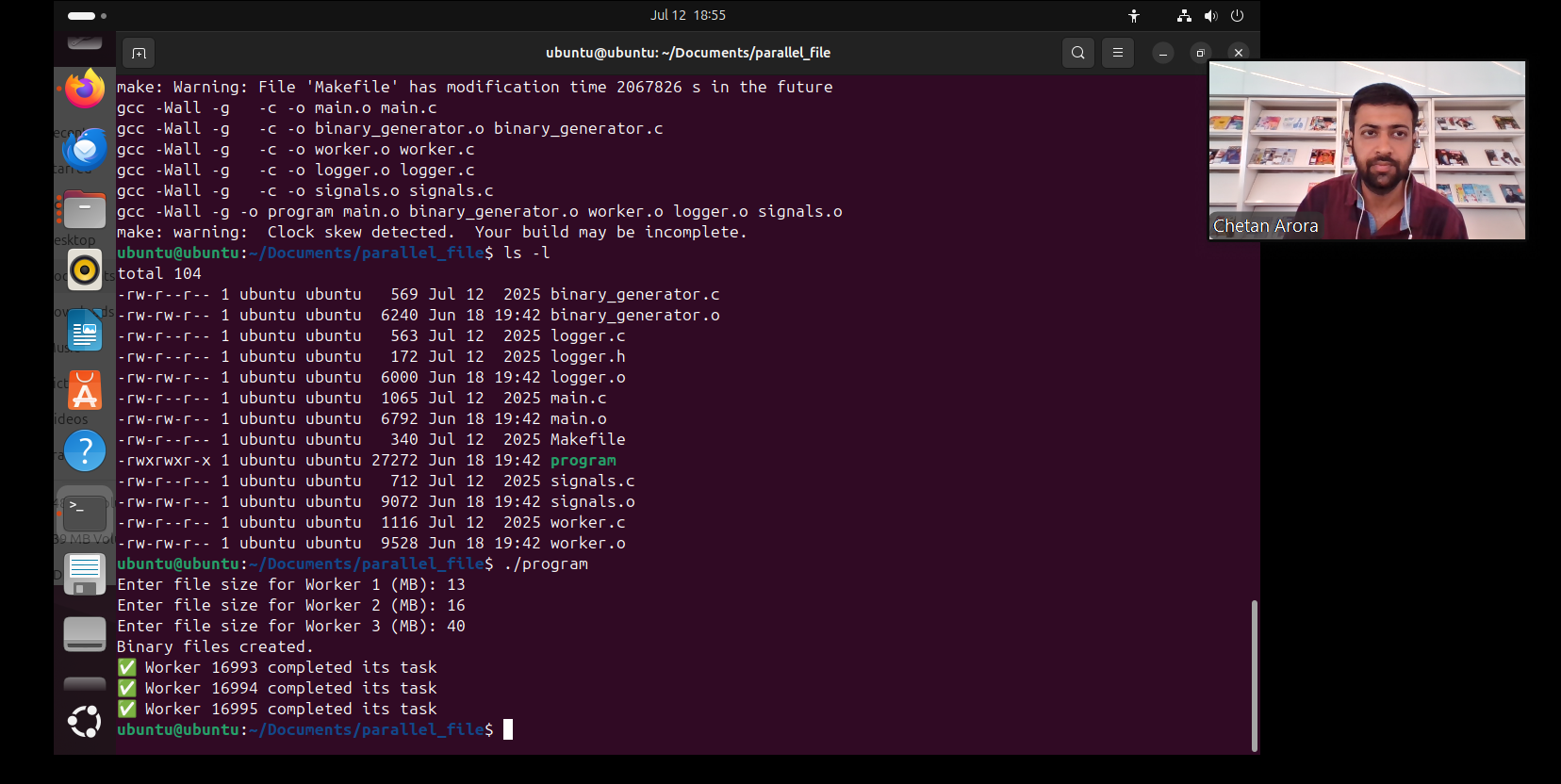
Ensuring that write operations are exclusive—only one writer at a time modifies the file.

Preventing interleaved writes, which could happen if multiple processes write partial messages at the same time.

This is crucial for logging systems, where message integrity and order are important.







Please answer the following two declarations:

o D1) On a scale from 1 to 5, How much did you use generative AI to complete this assignment?

▪

where:

▪

1 means you did not use generative AI at all

▪

2 means you used it very minimally

▪

3 means you used it moderately

▪

4 means you used it significantly

▪

5 means you relied on it almost entirely

▪

Your answer :2

o D2) On a scale from 1 to 5, How confident are you in your understanding of the generative AI support you utilized in this assignment, and in your ability to explain it if questioned?

▪

where:

▪

1 means "Not confident at all – I do not understand the generative AI support I used and cannot explain it."

▪

2 means "Slightly confident – I understand a little, but I have many uncertainties."

▪

3 means "Moderately confident – I understand the majority of the support, though some parts are unclear."

▪

4 means "Very confident – I understand most of the AI support well and can explain it with minor gaps."

▪

5 means "Extremely confident – I fully understand the generative AI support I used and can clearly explain or justify it if asked."

▪

Your answer :5