CPSC 457 – Fall 2025

Assignment 1: Processes in Linux — Marking Rubric

A. Code Portion (Total: 90 points)

Requirements:

- Your code must compile and run on **cslinux.ucalgary.ca**.
  - If your code does **not run or compile** on the university server, you will receive **zero** for the code portion.
- Your TA will check your GIT repository for version history.
  - Assignments with **no version history on GIT will receive zero** for the code portion.

Part 1 Treasure Hunt (40 points):

Criteria	**Points**
Correct Output and Format: - Output matches the required format. - Correct values are reported.	25
Process Management: - Correct use of fork (100 child processes, one per row). - Proper use of exit() and wait() for communication. - All child processes are properly terminated; no zombie processes.	5
Input Handling and Error Checking: - Properly reads and parses the input file. - Handles edge cases (e.g., no treasure in matrix, invalid input).	5
Efficiency: - The program runs efficiently and completes execution promptly. - There are no unnecessary delays or performance bottlenecks.	3
Code Quality: - The code is clean, well-organized, and easy to read. - Variable and function names are meaningful and descriptive. - Clear and meaningful comments are included throughout.	2
Total	**40**

Part 2 Parallel Prime Finder (50 points):

Criteria	**Points**
Correct Output and Format: - Output matches the required format. - Correct values are reported.	25
Process Management: - Correct use of fork() to spawn N child processes. - Proper use of exit() and wait() for process control. - All child processes are properly terminated; no zombie processes.	5
Shared Memory Management: - Correct creation and attachment of the shared memory segment using system calls (shmget, shmat, shmdt, shmctl). - Proper detachment and removal of shared memory after use.	5
Memory Layout and Range Assignment: - Each child writes only to its assigned range in shared memory (e.g., using a MAX_PRIMES_PER_CHILD value). - The memory layout ensures no overlapping writes or race conditions, and the parent correctly reads all assigned blocks.	5
Input Handling and Error Checking: - Properly parses command-line input. - Handles edge cases (e.g., invalid input, range smaller than N).	5
Efficiency: - The program divides work evenly among children and runs efficiently, even for large input ranges. - No unnecessary delays or performance bottlenecks.	3
Code Quality: - The code is clean, well-organized, and easy to read. - Variable and function names are meaningful and descriptive. - Clear and meaningful comments are included throughout.	2
Total	**50**

B. Reflection Portion (Total: 10 points)

Requirements:

- Reflection should be included in the submitted PDF file (2 points).

Part I:

- Discusses challenges such as reporting the treasure's location without shared memory, linking each child process to its row, and handling column numbers too large for exit codes (2 points).

Part II:

- Explains how the number of child processes is controlled (not exceeding N) (2 points).
- Describes the division of the input range among processes (2 points).
- Discusses the approach for safe access to shared memory by each child (2 points).