YEDITEPE UNIVERSITY

CSE331 OPERATING SYSTEMS DESIGN
FALL 2024

ASSIGNMENT I

Last Submission Date:
4 November 2024, 23:59

UNIX PROCESSES AND POSIX THREADS

In this assignment, you will explore and compare three programming paradigms: multi-process, multithreaded, and single-process programming. You will develop one multi-process program, two multi-threaded programs, and one single-process program. Each program will perform updates on a long array containing **50 million random positive integers**.

The array will be divided into segments (subarrays) for the update operation, as illustrated in Figure 1. In this example, with a subarray size (n) of 5 and a total of 50 million elements (N), there will be (N/n) 10 million subarrays created (as shown in Fig 1).

For this assignment, the subarray size (n) should range between 1000 and 1000000. This means your program will divide the array into anywhere from **50 to 50000 subarrays**.

Specific implementation details for each program are provided below.

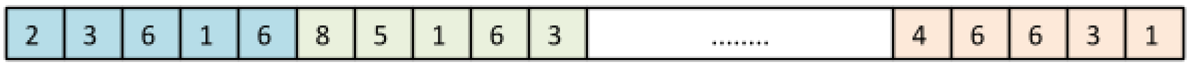

Figure 1

IMPLEMENTATION (50 pts)

Choose a subarray size n (1000 to 1000000). And write your programs accordingly.

a) Multi process program (20 pts): Develop a **multiprocess** program that creates child processes, with each child responsible for incrementing elements in its designated subarray of the long array. The parent process should create (N/n) child processes at the beginning of the program. Each child process will execute its update operation concurrently. At the end of the execution, the parent process should print all the updated elements of the long array and calculate the total execution time.

As you already know, in a multiprocess program, the parent process cannot see the changes made in child processes, so you need to **find a way** to propagate those changes back to the parent.

Write your code in **hw1a.c**.

b) Multi Threaded Program With Mutual Exclusion (15 pts): Create a **multithreaded** program that spawns threads, with each thread incrementing elements in its assigned subarray. The main thread should create (N/n) threads for the update operation. Each thread will perform its update concurrently, with synchronization **enforced** using **mutual exclusion** for each update operation. After all updates, the main thread should print the updated elements of the long array and calculate the total execution time. Write your code in **hw1b.c**.

c) Multi Threaded Program Without Mutual Exclusion (10 pts): Implement a **multithreaded** program that creates threads to increment elements in the subarrays of the long array. The main thread will create (N/n) threads for the update operation, with each thread executing concurrently. In this case, perform the updates **without using mutual exclusion**. After completion, the main thread should print the updated elements of the long array and calculate the total execution time. Write your code in **hw1c.c**.

d) Single Process Program (5 pts): Write a **single-process** program that performs the entire update operation without creating any child processes or threads. After the updates, it should print all the updated elements of the long array and calculate the total execution time. Write your code in **hw1d.c**.

REPORT (50 pts)

You are required to write a report detailing the design, implementation, and execution flow of your programs, as well as your insights regarding processes and threads. Using **<u>total execution times</u>** (not CPU time) from each program, compare the performance of multiprocess versus multithreaded programming. Discuss how variations in the total execution time are influenced by different values of **n** (subarray size). **Choose a small, medium and large value of n and try to evaluate the effect of subarray size on the total execution time.** Are the results in sync with the expected outcome?

Additionally, compare these programming paradigms using another metric that you consider crucial for illustrating the differences. Clearly explain your results and support them with graphical representations in your report.

SUBMISSION RULES

- **Writing clean, readable code and using comments are recommended.**
- **Include the student name and ID at the top of the code, as comments are required.**
- **Filenames of the C files must be hw1a.c, hw1b.c, hw1c.c, and hw1d.c.**
- **For report files, only PDF format is accepted. The filename of the report file must be Report1.pdf**
- **This homework will be sent to YULEARN.**

- **The submission must be zipped only as a zip file, no other zip formats will be evaluated.**
- **The name of the zip file should consist of student ID, assignment type, and number, excluded from space and special characters.**
- **Your C files will be checked for plagiarism using the MOSS system.**
- **Do not send your files via email. The files sent by email will not be taken into account!**
- **Any copied work (including ChatGPT etc. generated text or code!!!) will be evaluated as 0 points and the assignment will not be accepted!!!**
- You are welcome to ask questions and come up with new ideas about the homework, but reading the instructions explained here carefully, and studying from the course book are highly recommended to have a general understanding before asking questions. More creative ideas can get higher points.