1 Home Assignment 1: Parallel File Search

Objective: Use threads to search for a keyword in multiple files concurrently.

Instructions:

- Create a thread for each file provided via command line.
- Each thread searches its file for a specific keyword.
- Print the name of any file(s) that contain the keyword.
- Use mutexes if printing from multiple threads.

Expected Skills: Argument handling, thread creation, file I/O, mutex for safe console output.

---

2 Home Assignment 2: Threaded Matrix Row Sum

Objective: Compute the sum of each row in a matrix using threads.

Instructions:

- Accept a matrix of integers (e.g., 10×10).
- Create one thread per row.
- Each thread computes the sum of its row and stores it in a result array.
- Display all row sums at the end.

Expected Skills: Data structures, array indexing, thread coordination, dynamic memory allocation.

---

3 Home Assignment 3: Thread-Safe Logger

Objective: Implement a thread-safe logging system.

Instructions:

- Multiple threads write log messages to a shared file.
- Implement a log_message(const char*) function.
- Ensure no log entries overlap or interleave.

- Add timestamps to each log entry.

Expected Skills: File I/O, mutexes, system time functions, real-world synchronization.

---

4 Home Assignment 4: Multi-threaded Sorting (Chunked Merge Sort)

Objective: Sort a large array using a basic multi-threaded merge sort.

Instructions:

- Split the array into N chunks.
- Create one thread per chunk to sort it using qsort().
- In the main thread, merge the sorted chunks into a final sorted array.
- Measure and display time taken.

Expected Skills: Thread orchestration, sorting algorithms, timing functions, merging logic.

---

5 Home Assignment 5: Rate-Limited Counter with Condition Variables

Objective: Implement a shared counter that is increased by multiple threads, but **only 1 increment per second** globally.

Instructions:

- Create multiple threads that want to increment a shared counter.
- Use pthread_cond_t and pthread_mutex_t to ensure **only one increment per second**, regardless of how many threads are running.
- Run for 10 seconds and print the counter value at the end.

Expected Skills: Advanced synchronization, timing, condition variables, rate-limiting logic.