Assignment - 06
CS341: Operating System Lab

General Instruction

- Markings will be based on the correctness and soundness of the outputs. Marks will be deducted in case of plagiarism.
- Proper indentation & appropriate comments (if necessary) are mandatory in the code.

1 Problem Statement

In today's lab assignment, you are required to implement a concurrent matrix multiplication program in C using the concept of multithreading. The goal is to multiply two square matrices of size $N \times N$ efficiently using threads. Each thread will be responsible for computing a single element of the result matrix.

Requirements:

Matrix Input:

- Prompt the user to input the size N of the square matrices (e.g., $N = 3$ for a 3×3 matrix).
- Prompt the user to input the elements of the first matrix A and the second matrix B.

Thread Creation:

- Create $N \times N$ threads where each thread computes a single element in the result matrix C.
- The element $C[i][j]$ is calculated as:

$$C[i][j] = \sum_{k=0}^{N-1} A[i][k] \times B[k][j]$$

- Ensure that each thread correctly calculates the corresponding element $C[i][j]$.

Synchronization:

- Use appropriate synchronization mechanisms (e.g., mutexes or semaphores) to ensure that all threads are properly synchronized, especially if shared resources (e.g., the result matrix) are accessed concurrently.

Error Handling:

- Implement error handling for any potential issues during thread creation or synchronization. For example, if a thread fails to be created, the program should handle the error gracefully and terminate with an appropriate error message.

Output:

- After all threads have completed their computations, print the resultant matrix C to the console.
- Ensure the output is well-formatted and clearly indicates the result matrix.

Example:

For a 2×2 matrix multiplication, where:

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \quad B = \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}$$

The resultant matrix C should be:

$$C = \begin{pmatrix} 19 & 22 \\ 43 & 50 \end{pmatrix}$$

2 Problem Statement

Implement a concurrent file search program in C using the concept of multithreading. The goal is to search for a specific keyword in multiple files efficiently by dividing the search task among threads. Each thread will be responsible for searching through a specific file.

Requirements:

File Input:

- Prompt the user to input the number of files to search through.
- Prompt the user to input the names of the files and the keyword to search.

Thread Creation:

- Create a thread for each file. Each thread should search for the keyword in the corresponding file.
- If the keyword is found, the thread should return the line number(s) where the keyword appears.

Synchronization:

- Use appropriate synchronization mechanisms to ensure that the search results are printed in the correct order.
- Ensure that shared resources (e.g., console output) are accessed properly to avoid race conditions.

Error Handling:

- Implement error handling for any potential issues during file opening, thread creation, or synchronization. If a thread fails to be created or a file fails to open, handle the error gracefully and terminate with an appropriate error message.

Output:

- After all threads have completed their search, print the filenames and the corresponding line numbers where the keyword was found.
- Ensure the output is well-formatted and clearly indicates the results.

Example:

For three files with content:

```
file1.txt:
Hello world
This is a test
Keyword is here

file2.txt:
Another test
Keyword is also here
More content

file3.txt:
No keyword here
Just some text
```

If the keyword is "Keyword," the output should be:

```
file1.txt: Line 3
file2.txt: Line 2
```

Submission:

Submit your C program code along with a detailed explanation of the synchronization mechanisms you used. Make sure to include comments in your code to explain the logic, particularly the multithreading and synchronization parts.