Machine Problem 3: Process Synchronization & Communication

The Assignment

In this machine problem, you are to develop some simple synchronization and communication programs to better understand these topics. You have two tasks:

- Task 1: Synchronization. Based on the multi-threaded file copy tool (`mcopyfile`) you have created in Lab 2, now please use a worker pool (*Producer-Consumer model*) implementation that uses a fixed number of threads to handle the load (regardless how many files in the directory to copy). Your program should create 1 *file copy producer* thread and multiple *file copy consumer* threads (this number is taken from the command-line argument). The *file copy producer* thread will generate a list of (source and destination) file descriptors in a buffer structure with bounded size (refer to Textbook Chapter 16.7, page 573, for more detailed description on the possible data structure to use). Each time when the producer accesses the buffer it will write one (source, destination) file entry (per visit). And all *file copy consumer* threads will read from this buffer, execute the actual file copy task, and remove the corresponding file entry (each consumer will consume one entry each time). Both producer and consumer threads will write a message to standard output giving the file name and the completion status (e.g., for producer: "Completing putting file1 in the buffer", for consumer: "Completing copying file1 to …").
    - Please remember that there are several critical sections in this problem, e.g., accessing the buffer structure, writing messages to standard output. You need to use synchronization mechanisms we have learned in the class to protect these critical sections and synchronize threads.
    - Please read the text book, Chapter 16.7, page 573-575, for more detailed information on this task.
    - Measure the time before the first thread is created and after the last join (i.e., all threads finish the file copying). Display the

total time to copy the files in the directory. (You can use the function `gettimeofday` to measure time.)
    - Experiment with different buffer sizes and different numbers of consumer threads. Which combinations produce the best results? Try to explain.

- Task 2: Inter-Process Communication. Write a program to compare the performance of Pipes, Named Pipes (FIFO queues), and message queues. Measure the time to transmit data items of 1kB, 2kB, 4kB size between two processes using these different IPC mechanisms. Repeat the measurements sufficiently often to achieve a reasonable confidence in your results. Explain your results and conclusion. (You can use the function `gettimeofday` to measure time.)

What to Hand In

- Your programs in both tasks.
- A detailed report that describes your design, results, findings, analysis, and thoughts. It should contains answers to all the questions above, detailed procedure with necessary screen captures, issues/problems encountered and how you fix them, understanding of systems after the lab, and any lesson you have learned and thoughts you have about the lab.