CS307 Systems Practicum
Assignment 4
Processes

1.**Dining philosophers problem:**

5 Philosophers sit around a table and eat maggi using pair of forks. Each philosopher has a left fork and a right fork (A total of 5 forks on the table). Job of philosophers is to eat and think repeatedly but one can eat only when both of his left and right forks are available. A philosopher can't eat for more than 1 sec at a time and then he must release both of his forks to the table and think for some random time and then again try to eat.

Use threads with locks to simulate the philosophers' eat think life-cycle. Print state of each philosopher (thinking/eating/forks acquired etc.) after each change. Does your solution allow for fair and even eating by each philosopher?
(Keywords : dining philosophers problem, deadlock, starvation)

2. Four thread synchronization :

Write a program to create three threads T1,T2, T3 and T4.
Job of T1 is to take input as an integer and a string with space between them from the keyboard.
Job of T2 is to encrypt that input string and print it on display.
Job of T3 is to print nth fibonacci number on the display.
Job of T4 is to print nth fibonacci number and encrypted string with space between them.
Encryption is done using encryption table (as you have done in assignment 3). All threads should be synchronized ie the order input-encryption-fibonacci no.-display-input should be maintained.

3.Matrix Multiplication :

Write a sequential program to multiply two square matrices. You can initialize matrices with any random values, size of matrix (n) should be taken as command line argument. Now create a parallel program using threads to do the same. Run both program for different values of n (n=1 to at least 3000) and make proper log of respective running time. Plot graph of running times vs input size for both programs. (use dynamic memory allocation for arrays and avoid running any heavy program at the time of execution.)

Note:
Use pthread (POSIX Thread) libraries in C. Before attempting anyone of these problem first refer to basic thread programming examples available on the Internet. For thread synchronization use mutex variables (pthread_mutex) you can also use semaphores. For problem 3 make graph from log file only and plot both cases on single graph with proper labeling..

Please learn to use makefile and use comments and documentation in your code.