Programming Languages Lab Assignment-1
Bhavya Bansal- 150101015
Shradha Pruthi- 150101070

Problem 1 – Sock Matching

1. The role of concurrency and synchronization

>> Concurrency :
1. All robotic arms pick up socks simultaneously.
2. The matcher matches different colored socks simultaneously, thus increasing the processing speed.

>> Synchronization:
1. Two robotic arms don't pick up the exact same sock from the heap.
2. Matcher shouldn't get two alike colored socks simultaneously from two different robotic arms as it will give incorrect count. So matcher should access two same colored socks sequentially.

We made a random array having integers (1,2,3,4) representing all 4 colors of socks. We also made an array of size 4 with each index representing color, which is initialized to 0.
Then for each robotic arm, a separate thread is made. **This handles concurrency**.
Each arm chooses random index of the array and the number at that index represents color of the sock picked by the arm.
We ensure that no two robotic arms pick the same index by making reentrant lock for each index. **This handles synchronization.**
Now, the arm accesses the color array at the chosen color' s index. If it's 0, then we make it 1. Else, we make it 0 and increase the count of matched socks. We also make a reentrant lock for each color in color array. **This also handles synchronization.**

Problem 2 – Data Modification in Distributed System

1. Why concurrency is important here?
    a. Concurrency allows more than one record or more than one file to be updated by different people simultaneously.
    b. Thus it helps in making use of multi processing facilities and saves time. It increases the efficiency of the application.

2. Shared Resources :
    a. Shared resources in a multithreaded server are resources that can be accessed concurrently. In addition to scope object attributes, shared resources include in-memory data, such as instance or class variables, and external objects, such as files, database connections, and network connections.

    b. In our program, **the shared resources include the record or the file being accessed concurrently by multiple task threads.**

3. What may happen if synchronization is not taken care of? Give examples.
    a) If synchronisation is absent, then output of marks of a record being accessed simultaneously by two task threads may be inconsistent. When multiple threads try to access a shared resource concurrently, and these operations overlap in execution, then the final outcome depends on the order in which these operations take place, which is unpredictable. This phenomenon is called Race condition.

       Example :
       If the mark of Amit Kumar Sharma is modified from 75 to 80 by TA2 and TA1 want to decrease the mark by 3, the sorted files may contain either 77 or 80 or 72.

Memory inconsistency errors, i.e., different threads may have inconsistent views of the same data. This happens when one thread updates some shared data, but this update is not propagated to other threads, and they end up using the old data. It is resolved by using **volatile** keyword.
Example:
If the mark of Amit Kumar Sharma is modified from 75 to 80 by CC and TA1 wants to decrease the mark by 3 and we make sure to check thread consistency, that is only one thread accessing at one time, then the final marks can be 77 which is not possible as TA1 can’t modify the record after CC. The reason could be the last teacher record of Amit may not be updated for the other thread as CC, so TA1 can also modify it.

4. Concurrency is handled by making different threads for each task of updation. Synchronization for Race Condition is handled by synchronized blocks for record updation and on files in file level modification.

Synchronization for memory inconsistency is handled by **volatile** keyword on records and files.

Problem 3 – Room Delivery Service of Tea/Snacks.

Role of concurrency

Here, concurrency allows more than one client to put a request simultaneously and get the estimated time by making threads for each client.

Synchronisation is required as a client thread' s order should be delivered in first come first serve manner and the tea(or coffee) machine can't be accessed simultaneously by two threads as it brews one cup at a time.

Synchronisation is important as without it we may get incorrect delivery times.

Example:

If client 1 orders 5 cups of tea and client 2 simultaneously orders one tea, then if synchronisation is missing, they will access the tea machine concurrently , so the time for both will be 7 and 3 minutes respectively. But this is incorrect and with synchronization, if client 1 gets the tea machine first, then the respective times will be 7 and 8 minutes, which should be true as the tea machine can only be accessed sequentially.

Concurrency is handled by making threads for each client request.

Synchronisation is handled by making a lock with priority queue on each of tea machine, coffee machine and snacks.