Assignment 1 (Concurrent Programming) Group 13 Report

Team Members:
150101027 - I.N.Dilip Kumar
150101051 – Reddi Hareesh

Problem 1 – Sock Matching:

The role of concurrency and synchronization in the above system?
The concurrency aspect in the given problem is that all the Robotic Arms will select the socks at the same time.
All the robotic arms are using the same matching machine and also some shared variables like the Socks. This is where the synchronized role comes in. The socks are to be added to the matching machine in a synchronized manner and also the sock is to be accessed so that one object is not accessed by more than one thread.

How I handled those?
Concurrency is maintained by using threads i.e. one thread for each of the robotic arm. Each thread operating independent of other threads which is just like how the Robotic arms in the system works.
Synchronization is maintained by using the ***Reentrant lock*** for the Sock array and synchronized keyword for the matching machine. The locks are used so that at one point of time only one thread can use the Sock array to pick up the sock and any other thread that is trying to access the same object will get access denied.

Problem 2 – Data Modification in Distributed System:

Why concurrency is important here?
While we are doing file level modification the two changes made by the TA's or the CC must happen concurrently. So to do these changes concurrency is used.

What are the shared resources?
The file Stud_Info.txt and all the students objects formed using this information are also shared resources. Here each student object is made for each of the student details.

What may happen if synchronization is not taken care of? Give examples.
If synchronization is not taken care of then the marks in the file may not get updated in the fashion we want to be.
For example consider the initial Stud_Info.txt as follows

Roll No	Name	Mail_id	Marks	Teacher
174101055	Amit Kumar Sharma	amit55@iitg.ac.in	75	TA1
174101012	Abdul Najim	abdul12@iitg.ac.in	29	TA2
174101058	Kunal Kishore	Kunal58@iitg.ac.in	67	TA2
174101033	Subhra Shivani	subhra33@iitg.ac.in	53	CC
174101035	Savnam Khatam	savanam35@iig.ac.in	88	TA1

Consider the following inputs
TA1 174101058 1 5 (TA1 wants to increase the marks of 174101058 by 5)
TA2 174101058 2 4 (TA2 wants to decrease the marks of 174101058 by 4)

If synchronization is not taken care of then final marks can be any one of 72(by TA1) or 63(by TA2)
If synchronization is taken care of then final marks will be correctly changed to 68.

How you handled concurrency and synchronization?
We have used multi-threading to handle the concurrency. When synchronized keyword is not used in record level or file level then both modifications by the teachers in both levels will happen simultaneously.
We have used ***synchronized keyword*** on each student object for Record Level Modification So, only one thread can modify the same student record at a time which in turns helps in maintaining synchronization. And for File Level modification we have used the synchronized keyword on the entire Student class.

Problem 3 – Room Delivery Service of Tea/Snacks:

What role concurrency plays here?
At one time multiple orders are also accepted by the server. To maintain all those orders we need concurrency. Because all the orders are received by one thread for each client and it calculates the estimated time for this order and then sends the estimated time to the client.

Do we need to bother about synchronization? Why? Illustrate with example.
We have to think about synchronization because there is only one queue maintaining the orders. If two orders from clients arrive at the same time and while we are calculating the estimated time for each of the orders, we have to calculate the time by accessing the queue in a synchronized manner. In this way the estimated time is correctly calculated when multiple orders arrive at the same time.

Example:
Consider the following example:
The present state of the queue

Order 1 (ET1 = 10m)	Order 2 (ET2 = 15m)	Order 3 (ET3 = 20m)	Order 4 (ET4 = 30m)		

And now let's say two orders Order 5 and Order 6 with Processing times PT5 = 6m and PT6 = 4m came at the same time.
If synchronization is not used then the Estimated times will be ET5 = ET4 + PT5 + 2 = 38.
And ET6 = ET4 + PT6 + 2 = 36.
If synchronization is used then the Estimated times will be ET5 = ET4 + PT5 + 2 = 38.
And ET6 = ET5 + PT6 + 2 = 44.
So synchronization is important for this question and has to be managed properly.

How you handled both?

Threads are used to maintain concurrency. Each thread is going to accept an order through a socket and then input the order in to the shared queue among all the threads. And this thread will also calculate the estimated time for the order and send it back to the client.

Synchronization is maintained among all the threads using ***binary semaphores***. So at one point of time only one thread can access the queue and input its order into the queue.