# CS431: Programming Languages Lab (2020) Assignment 1(Concurrent programming)

#### Name - Rishi Pathak

#### Roll Number - 170101054

**Problem 1: Sock Matching Robot:**

1. *The role of concurrency and synchronization in the above system.*
2. The role of concurrency in this problem is that all the robot arms work together simultaneously to pick the socks. The role of synchronization comes into play where all the robots are using the same matching machine and also some shared variables like socks. The socks are to be added to the matching machine in a synchronized manner and also the socks are accessed such that one object is only accessed by one thread at a time.

*Q) How did you handle it?*

A) I made one thread for each robotic arm. These threads take care of the concurrency. Each thread operates independent of the other threads which is like how the Robotic arms in the problem worked. Synchronization is maintained by using the Reentrant lock for the Sock array and synchronized keyword for the matching machine. This lock is used because at any particular time, only one robot thread can use the Sock thread to pick up a sock. If any other thread tries to access the sock thread at that time, then access will be denied.

**Problem 2: Data Modification in Distributed System:**

*Q) Why concurrency is important here?*

A) When we do file level modification, the changes made by TA1, TA2 or CC must happen concurrently. So for doing this, concurrency is important here.

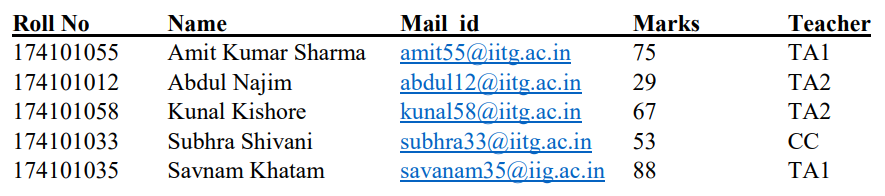
*Q) What are the shared resources?*

A) The resources that are shared here are the Stud\_Info.txt file and all the Student objects made by reading the file. The object store the information about the student that is a record in the file.

*Q) What may happen if synchronization is not taken care of? Give examples.*

A) If the synchronization is not taken care of then the writes will not performed in the correct order or the order that we expect them to. There may be race conditions also.

*Example-* Let us take the initial Stud\_Info.txt



Consider this input:

TA1 174101055 1 10 (TA1 wants to increase the marks of Amit by 10)

TA2 174101055 2 5 (TA2 wants to decrease the marks of Amit by 5)

If *synchronization* is not taken care of then, the final marks of Amit can be anything out of 85 or 70.

If *synchronization* is taken care of, the final marks of Amit will definitely be 80.

*Q) How you handled concurrency and synchronization?*

A) The concurrency is maintained using multi-threading. If synchronized keyword is not used in record-level or file-level then both modifications by the checkers in both levels will happen simultaneously. This will result in wrong results. I have used the *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 similarly for File Level modification, I have used the synchronized keyword on the entire Student class. Therefore, only one change happens at a time in the whole Stud\_Info.txt file maintaining the correctness.

**Problem 3: Calculator for Differently Abled Persons:**

*Q) How does the program work?*

*A)* I used the Swing for the implementation of the UI using Java, SwingWoker for concurrency, and Java KeyListner for the keyboard events.

Buttons need to be highlighted after specific time intervals. Timer cannot run on event dispatch thread because the UI will become unresponsive in between. Therefore the button presses and timer are made into different threads. SwingWorker is a multi-threading utility class for the Swing library, which enables proper use of the event dispatching thread. Also for more efficient program, thread is suspended for 750ms.