CS 558: Computer Systems Lab Assignment
Deadline: 4th April 2021

Things to note before you start:

- This assignment is to be done in groups. Each group should be of 2 students only. Choose your own partner for the assignment. Please do the same by 20th March 2021.
- No extensions in submission are allowed. Delay in submission will lead to penalty in marks.
- The programs can be written in C/C++/Java.
- Assignments submitted before the deadline will only be considered for evaluation.
- Please do not email your assignments separately to the TAs, it will not be considered for evaluation.
- Your code will be checked for plagiarism. Any kind of academic dishonesty, plagiarism, etc. will lead to penalties.
- NO sharing of code between students, submission of downloaded code (from the Internet, Campus LAN, or anywhere else) is allowed.
- The first instance of code copying will result in ZERO marks for the assignment.
- The second instance of code copying will result in a `F' grade. Students may also be reported to the Students Disciplinary Committee, which can impose additional penalties.

1. Implement a preemptive priority scheduling algorithm considering 9 processes in a system used by a student in the department. The system has a unique way of assigning priorities to the processes. The Roll Number of the student is used to assign priorities to the processes in the following way- i^{th} digit of the Roll number is the priority for the i^{th} process. For e.g.- if the roll number is 176101802, then priorities of first and second processes will be 1 and 7, respectively. The arrival time of each process is again the digits of the roll number but in reverse order (e.g.- for the same roll number, the arrival time of process 1 is 2 and for process 2 is 0 and so on). The burst time for each process is a random number (system generated) between (2-7) milliseconds.
You need to show the average waiting time for scheduling the processes and also a Gantt chart showing the sequence of schedule of the processes.

2. A highway has a single lane tunnel. However, due to construction on a nearby road, all vehicles have to use the tunnel. Due to the size of the tunnel, a vehicle can safely enter the tunnel if and only if there are no other vehicles entering the tunnel. To prevent accidents, sensors are installed at each end of the tunnel which notify a controller system about the arrival and departure of vehicles in the tunnel, in either direction. The controller uses the sensor input to control signal lights at either end of the tunnel.
Assumptions

- Each vehicle can be represented by a thread that calls Arrive() and Depart() functions in the controller and passes arguments to indicate direction of travel
- Arrive() can safely stop an arriving vehicle by changing the correct signal light to Red and block the calling thread.

Implement the above scenario, i.e., the controller program using mutexes and condition variables. Your solution should correctly handle rush hour, during which many vehicles approach the tunnel from both directions.

3. In a small village, the Government has established a COVID vaccination center. The clinic has only one Doctor who will give the vaccination. There is one chair for the doctor to sit and N chairs for waiting patients. Consider the following scenario:

i) If there is no patient in the clinic, the doctor simply sits on his chair and does his personal work.
ii) When a patient arrives, he/she interrupts whatever the doctor is doing.
Iii) If there are many patients, then the remaining patients wait. Patients are allowed to wait only if there are empty chairs where they can sit, if not, then they are asked to leave.

Implement the above using Monitors and consider the following conditions:

- One Doctor, many patients.
- A finite waiting queue.
- One Patient is treated at a time.
- Doctor does his own work when no patients are waiting.
- Patient leaves if clinic is full, i.e, no chairs are available to sit.
- Patient sleeps while waiting in queue for his/her turn.

4. Five philosophers regularly meet for dinner and to discuss the issues of the day. They always eat spaghetti and they always eat with two forks. The diners are seated at a round table with a fork between each plate. As a result, there are five philosophers with five forks. Each of the philosophers wishes to eat dinner and each of them needs two forks to accomplish this task. In this assignment, you will have to simulate each philosopher with an independent thread. Your goal is to write a system that allows each philosopher to eat — nobody is permitted to starve to death.

Initially the philosophers are thinking, after a certain amount of time the philosopher gets hungry and wishes to eat. Once eating is done, the philosopher then puts the fork down, so that they are available for another neighbour.

If this system is coded without thought, each philosopher will immediately pick up, say, their left fork followed by their right fork, thereby blocking others from eating and forming a deadlock condition. A more intelligent approach will be needed. Develop the code to implement the given scenario.

Your Output.txt file must contain sample runs that show the action of each philosopher at the table. Each philosopher/thread must announce when he has picked up a fork (and which one: right or left), when he has “eaten,” and when he has put down a fork. You should run the program long enough to demonstrate that each philosopher has had the opportunity to eat at least once, and that dining privileges have been distributed fairly.