Assignment 4 (based on concurrency)

(Note: This has two problem statements and both have to be solved. The evaluation will be done on 6th Nov afternoon. Any late submission will be penalized with 25% marks. If one has taken prior approval for genuine reason, one should mail the solution before the deadline and come for evaluation at the earliest possible time.)

Q1. **The Readers-Writers Problem.** Consider an application where multiple threads of a process wish to read (reader threads) and write (writer threads) some shared data. There can be two ways to design such systems, with *reader preference* (when a reader thread is accessing the data, additional readers are allowed to concurrently hold the lock with previous readers even if a writer thread is waiting for the lock) or *writer preference* (additional readers are not granted the lock when a writer is already waiting). Write a program to simulate above mentioned application with ***writer preference (****as reader preference code* has been discussed in the class). Use PTHREAD library to implement the synchronization. You can refer to below table to understand the rules of *writer preference*.

Thread accessing shared data	**New request**	**Preferred action**
None	Reader	Starts reading.
None	Writer	Starts writing.
Only reader/readers	Writer	Writer blocked until readers **currently accessing** the database are finished.
Only reader/readers	Reader	Starts reading if no writers are blocked. Otherwise reader will be blocked until all writers complete writing.
Writer	Writer	New writer is blocked. Only one writer may write the data at a time.
Writer	Reader	Reader blocked.
Writer accessing, but readers blocked	Writer	Writer is blocked, but should write before any blocked readers (no matter when they arrive) are allowed to read.

You can test your code on request mentioned in below table.

Thread	R1	R2	R3	W1	R4	R5	W2	R6	R7
Arrival time	0	2	4	5	6	7	7	8	9
Execution time	8	6	5	6	2	2	4	2	2

Q2. The Sleeping-Barber Problem. A barbershop consists of a waiting room with *n* chairs and the barber room containing the barber chair. If there are no customers to be served, the barber goes to sleep. If a customer enters the barbershop and all chairs are occupied, then the customer leaves the shop. If the barber is busy but chairs are available, then the customer sits in one of the free chairs. If the barber is asleep, the customer wakes up the barber. Write a program to coordinate the barber and the customers (each customer should be a different thread). Use PTHREAD library to implement the synchronization.

Consider the following issues:

a) Is there a way by which you can evaluate through execution of your program whether the three conditions of Critical Section are satisfied?

b) How would you incorporate priority amongst customers? Let us say that there are two groups: staff and student. A waiting student customer has priority over the waiting staff customer (*because the students are supposed to be more busy and work hard*). Therefore, a barber attends to the waiting student customer after finishing the work on present customer. Modify the code to include the priority condition also. Your program should run both with priority and without priority and must also address the problem of starvation.

c) Another modification is that there is more than 1 barber; say there are four barbers. Now modify the code to include this condition also. (*A good code would include the number of barbers as variable implying the barbershop could hire or fire some of the barbers).*