Q. IndianRail has decided to improve its efficiency by automating not just its trains but also its passengers. Each passenger and each train is controlled by a thread. You have been hired to write synchronization functions that will guarantee orderly loading of trains. You must define a structure struct station, plus several functions described below.

When a train arrives in the station and has opened its doors, it invokes the function station\_load\_train(struct station \*station, int count)

where count indicates how many seats are available on the train. The function must not return until the train is satisfactorily loaded (all passengers are in their seats, and either the train is full or all waiting passengers have boarded).

When a passenger arrives in a station, it first invokes the function station\_wait\_for\_train(struct station \*station)

This function must not return until a train is in the station (i.e., a call to station\_load\_train is in progress) and there are enough free seats on the train for this passenger to sit down. Once this function returns, the passenger robot will move the passenger on board the train and into a seat (you do not need to worry about how this mechanism works). Once the passenger is seated, it will call the function

station\_on\_board(struct station \*station)

to let the train know that it's on board.

Create a file IndianRail.c that contains a declaration for struct station and defines the three functions above, plus the function station\_init, which will be invoked to initialize the station object when IndianRail boots. In addition:

You must write your solution in C using locks and condition variables:

 lock\_init (struct lock \*lock)



 lock\_acquire(struct lock \*lock)



 lock\_release(struct lock \*lock)



 cond\_init(struct condition \*cond)



 cond\_wait(struct condition \*cond, struct lock \*lock)

 cond\_signal(struct condition \*cond, struct lock \*lock)



 cond\_broadcast(struct condition \*cond, struct lock \*lock)

Use only these functions (e.g., no semaphores or other synchronization primitives).

 You may not use more than a single lock in each struct station.



 You may assume that there is never more than one train in the station at once, and that all trains (and all passengers) are going to the same destination (i.e. any passenger can board any train).



 Your code must allow multiple passengers to board simultaneously (it must be possible for several passengers to have called station\_wait\_for\_train, and for that function to have returned for each of the passengers, before any of the passengers calls station\_on\_board).



 Your code must not result in busy-waiting.