LOVELY PROFESSIONAL UNIVERSITY

CA-2

Course Title: Operating Systems

Max.Marks:30 ***DOS: 30th March, 2020***

By

Prabin Kumar Das

Reg. No: 11715435

Roll No: B-69

Section: EE032

GitHub Link: https://github.com/Prabindas001

GitHub Repository: Indian_railways
Email Address: Prabindas786@gmail.com

Submitted: Dr. Aarti

SEEE

Lovely Professional University
Phagwara, Punjab

Question:

6. Indian Rail 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 Indian Rail boots.

Code

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<pthread.h>

int num;                        //number of passengers present at the platform
struct station
{

  pthread_mutex_t tpLock;
  pthread_cond_t trainArrived;
  pthread_cond_t passengerSettled;
  int boarded_passengers;       //passengers boarded into the train
  int passengers_inStation;     //passengers waiting in the platform
  int seats_vacant;             //vacant seats in the train

};
//FunctionDeclaration
int min (int x, int y);
void station_init (struct station *station);
void station_load_train (struct station *station, int count);
```

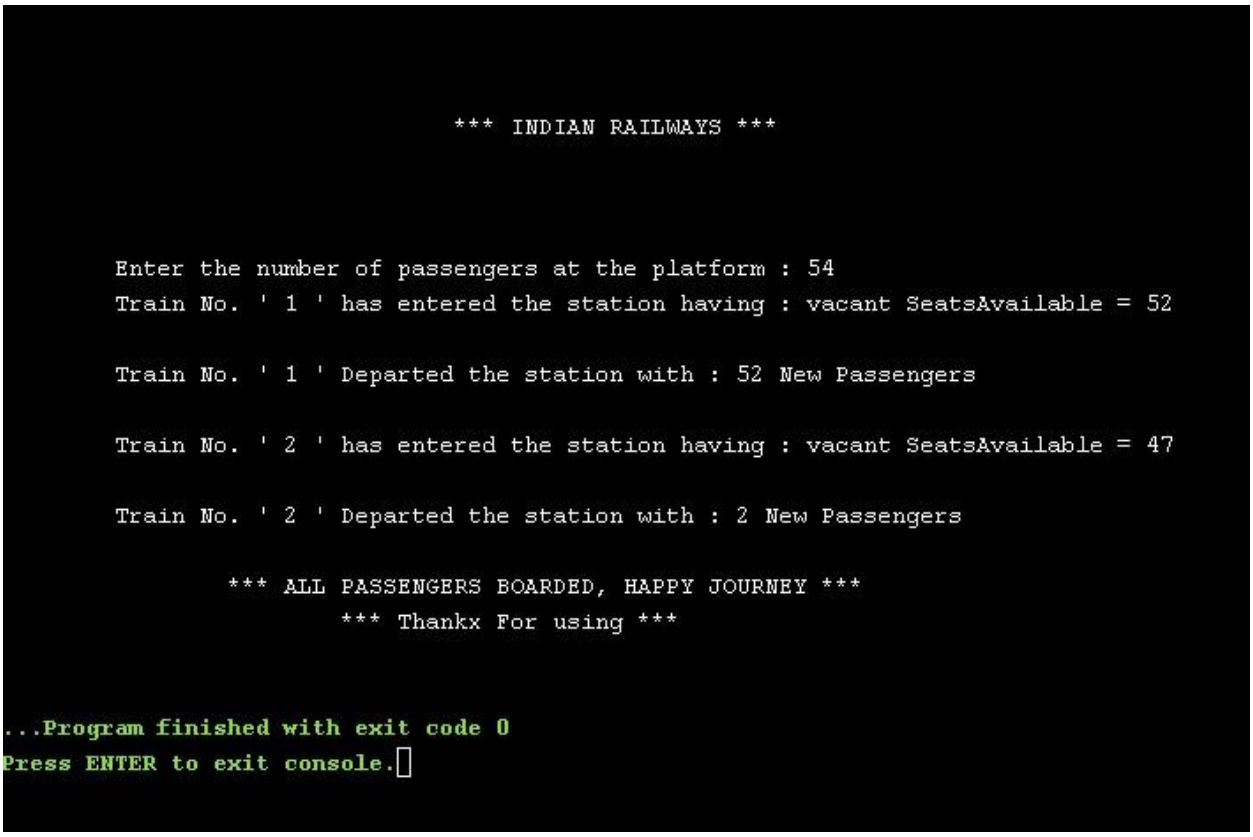
```
void station_wait_for_train (struct station *station);
void station_on_board (struct station *station);

//FunctionDefinition
Void station_init (struct station *station)
{

  pthread_mutex_init (&station->tpLock, NULL);     //initialize mutex locks
  pthread_cond_init (&station->trainArrived, NULL);          //thread condition variable
  pthread_cond_init (&station->passengerSettled, NULL);
  station->boarded_passengers = 0;
  station->passengers_inStation = 0;
  station->seats_vacant = 0;

}

/*train arrives*/
Void station_load_train (struct station *station, int count)
{
  //returns when there are no passengers or train is full

  pthread_mutex_lock (&station->tpLock);
  station->seats_vacant = count;
  while (station->seats_vacant > 0 && station->passengers_inStation > 0)
    {

      pthread_cond_broadcast (&station->trainArrived);        //similar to used signal and used to
inform several threads which are waiting
      pthread_cond_wait (&station->passengerSettled, &station->tpLock);

    }

  station->seats_vacant = 0;
  pthread_mutex_unlock (&station->tpLock);
}


//passenger arrives

Void station_wait_for_train (struct station *station)
{
  //return when there are enough available seats and train is in the station
  pthread_mutex_lock (&station->tpLock);
  station->passengers_inStation++;

  while (station->boarded_passengers == station->seats_vacant)
    {
      pthread_cond_wait (&station->trainArrived, &station->tpLock);
    }
```

```
  station->boarded_passengers++;
  station->passengers_inStation--;
  pthread_mutex_unlock (&station->tpLock);
}



//passenger boarded

Void station_on_board (struct station *station)
{
  //to inform the train that it is on board
  pthread_mutex_lock (&station->tpLock);
  station->boarded_passengers--;
  station->seats_vacant--;

  if ((station->seats_vacant == 0) || (station->boarded_passengers == 0))
    {
      pthread_cond_signal (&station->passengerSettled);
    }

  pthread_mutex_unlock (&station->tpLock);

}

volatile int threads_completed = 0;
void *passenger_thread (void *arg)
{
  struct station *station = (struct station *) arg;
  station_wait_for_train (station);
  threads_completed++;
  return NULL;
}

struct TrainLoaded_Para
{
  struct station *station;
  int free_seats;
};

volatile int return_LoadTrain = 0;

void *load_train_thread (void *args)
{
  struct TrainLoaded_Para *temp = (struct TrainLoaded_Para *) args;
  station_load_train (temp->station, temp->free_seats);
  return_LoadTrain = 1;
  return NULL;
}
```

```
//finds the minimum value among x and y
#ifndef MIN
#define MIN(_x,_y) ((_x) < (_y)) ? (_x) : (_y)
#endif
//main function starts from here
int
main ()
{
  struct station station;
  station_init (&station);

  srandom (getpid () ^ time (NULL)); //generates random numbers
  int i;
  printf ("\n\n\n\t\t\t\t*** INDIAN RAILWAYS ***\n\n");
//   printf
//     ("\n\t\tNOTE*:NUmber of free seats in each train is initialized to 60");
  printf ("\n\n\tEnter the number of passengers at the platform : ");
  scanf ("%d", &num);
  if (num < 0)
    {
      printf
           (" \tYou have entered '%d' passengers which is Invalid.\n",
            num);
      printf (" \tPlease enter a valid number!!\n");
      scanf ("%d", &num);
    }
  if (num == 0)
    {
      printf (" \t STATION IS EMPTY!!\n\n");
      return 0;
    }
  const int total_Passngrs = num;
  int remaining_Passngrs = total_Passngrs;
  for (i = 0; i < total_Passngrs; i++)
    {
      pthread_t tid;
      int ret = pthread_create (&tid, NULL, passenger_thread, &station);
    }

  int total_Passngrs_boarded = 0;
  const int tot_FreeSeats_PerTrain = 100;
  int pass = 0;
  int j = 1, p = 1;
  while (remaining_Passngrs > 0)
    {

      int free_seats = random () % tot_FreeSeats_PerTrain;

      printf
           ("  \tTrain No. ' %d ' has entered the station having : vacant SeatsAvailable = %d\n\n",
```

```
            j, free_seats);
      j++;
      return_LoadTrain = 0;
      struct TrainLoaded_Para args = { &station, free_seats };
      pthread_t lt_tid;
      int ret = pthread_create (&lt_tid, NULL, load_train_thread, &args);
      if (ret != 0)
           {
            perror ("pthread_create");
            exit (1);
           }

      int threads_to_reap = MIN (remaining_Passngrs, free_seats);
      int threads_reaped = 0;

      while (threads_reaped < threads_to_reap)
           {

            if (return_LoadTrain)
              {
                exit (1);
              }
            if (threads_completed > 0)
              {
                if ((pass % 2) == 0)
                     usleep (random () % 2);
                threads_reaped++;
                station_on_board (&station);
                threads_completed++;

              }
           }

      remaining_Passngrs -= threads_reaped;
      total_Passngrs_boarded += threads_reaped;
      printf
           ("  \tTrain No. ' %d ' Departed the station with : %d New Passengers\n\n",
            p, threads_to_reap);

      pass++;
      p++;
    }

  if (total_Passngrs_boarded == total_Passngrs)
    {
      printf ("\t\t*** ALL PASSENGERS BOARDED, HAPPY JOURNEY ***\n");
      printf ("\t\t\t*** Thankx For using ***\n");
      return 0;
    }
}
```

Output:


*** INDIAN RAILWAYS ***
Enter the number of passengers at the platform : 54
Train No. ' 1 ' has entered the station having : vacant SeatsAvailable = 52
Train No. ' 1 ' Departed the station with : 52 New Passengers
Train No. ' 2 ' has entered the station having : vacant SeatsAvailable = 47
Train No. ' 2 ' Departed the station with : 2 New Passengers
*** ALL PASSENGERS BOARDED, HAPPY JOURNEY ***
*** Thankx For using ***
...Program finished with exit code 0
Press ENTER to exit console.