McGill University
ECSE 427 / COMP 310

Programming Assignment #2: Synchronization Winter 20018

Due date: Check My Courses
Drafted by: Dimitrios Gallos and Dr. Rola Harmouche

The Pierre Elliot Trudeau Airport has hired you to simulate the usage of Taxis by travellers arriving from their flights. In particular, they want to investigate how many Taxis are needed for a given number of airplane arrivals each hour. They estimate that from each plane, 5-10 people take a Taxi. This number is sampled from a uniform distribution (hint: use the function rand()).

Passengers wanting to take a taxi arrive on the waiting platform at the beginning of each hour (regardless of when the plane actually landed). When passengers arrive on the waiting platform, they form a queue (hint: implement this as a queue). Airport rules allow only 100 people in the queue for safety reasons. If there is no room in the queue, passengers get impatient and take another form of transportation. Taxis take between 10 minutes to 30 minutes to take the passengers to their destination and come back to the airport, again modelled by a uniform distribution. For simplicity, assume people enter taxis individually. If there are no people waiting for taxis, the taxis wait.

Your job is to write a program to model this process. In particular, you should model the planes arriving and the taxis leaving as threads (one thread for each taxi and one thread for each airplane). Ensure all the requirements mentioned above are met by using synchronization tools from the pthreads library. Model the timings mentioned using seconds, i.e. 1 hour should be simulated as 1 second (hint : use the function sleep()).

Your program should be run with the following parameters:

```
>./air_taxi_sim num_airplanes num_taxis
```

The events you are required to log are : Which airplane's passengers did not go to the platform because it was full, which airplane's passengers did get to the platform, which taxi driver is waiting, which taxi driver picked up which passengers.

For printing purposes name the taxis from 0 to N-1 where N is the number of taxis. Name the airplanes from 0 to M-1 where M is the number of airplanes. Each passenger should get an ID as well with the following convention : 1ZZZYYY, where ZZZ are 3 digits designating the plane number from which the passenger arrived, and YYY are 3 digits designating the passenger number. (clearly we cannot have more that 1000 planes and 1000 passengers). Given this convention, the names of passengers are repeated every hour, i.e. the first passenger of the first plane arriving at hour 0 has the same ID as the first passenger of the first plane arriving at hour 1.

A sample printout for each case should look like the following:

```
You entered: 3 airplanes per hour
You entered: 10 taxis
Creating airplane thread 0
```

```
Creating airplane thread 1
Creating airplane thread 2
Airplane 0 arrives with 8 passengers
Passenger 1000000 of airplane 0 arrives to platform
Passenger 1000001 of airplane 0 arrives to platform
Taxi driver 0 arrives
Taxi driver 0 picked up client 1000000 from the platform
Taxi driver 1 arrives
Taxi driver 1 picked up client 1000001 from the platform
…

Platform is full: Rest of passengers of plane 2 take the bus
…
Taxi driver 8 waits for passengers to enter the platform
…
```

The last statement is printed in the case that the platform is full of passengers.

Note: Think of this assignment in terms of the producer-consumer problem. However, there is 1 major issue to consider which affects the implementation: When the queue is full, passengers do not wait on it to empty before entering.

What to hand in:

You should hand in 1 c file which implements the whole process, and a report.

Include the following in your report:

1) Explain why you need to use synchronization tools to properly model this process. What would happen if you did not use synchronization tools (i.e. describe a problem that would occur each time a synchronization primitive is omitted)?

2) Experiment with [2, 5, 10] airplanes arriving each hour and [100, 10, 2] taxis. Print out what each taxi and airplane is doing and include that print-out in your report. Let your program run for 10 “hours” (airplane arrivals).

Can be 5 hours

3) Describe any issues you may have in your code which you believe is making it run incorrectly (i.e. any known bugs). This would you get partial marks in case your code is not fully functional.

Submission guidelines will be posted in the submissions folder on MyCourses.