Implementation of Ober Cab Services

Basic design of code:

- For each cab creating a mutex lock.
- For each rider getting max_wait time ,ride time and cab type.
- Creating two semaphores one having value number of cabs and other having number of payment servers.
- Creating a thread for each rider which calls the booking() function and waiting for random time for next rider.
- Waiting for all the threads of the riders to get completed.

Explanation of functions:

payment():

If a rider is done with the ride , it checks for a server which is free and do its payment .

booking():

Here there are two cases:

If the rider needs a premier cab:

First getting the current time and adding max_wait to the current time. And now passing this time to the sem_timedwait() to check if a cab will become free before the max_wait time. Them checking which cab is free to serve and start riding in it.

If the rider needs a pooling cab:

I this case i am doing pooling because just using semaphores wont work here.

From current time till the max_wait time if there is any cab available then there are two things.

If there is an empty cab available :

Doing the same thing as in case of premier cab.

If there is a cab with one passenger in it:

Just incrementing the cab count by one.

Things in mind while coding:

If cab[i]=0 means cab is empty

If cab[i]=1 means it is serving a premier rider

If cab[i]=2 means it is a pooling cab with one passenger in it.

If cab[i]=3 means it is a pooling cab with 2 passengers in it.

If there are two riders in a cab then the cab will become free after both riders have spent their ride time . **NOTE:** some part of their ride time can overlap.