Problema:

Un **servizio di pool-car** ha a disposizione N automobili, ognuna assegnata ad una zona della città. Gli utenti del servizio si recano al punto di partenza e prenotano un posto sull'automobile per la zona desiderata. Ogni automobile ha a disposizione M posti. Una volta occupati gli M posti, il veicolo parte per la zona stabilita. Nel caso in cui sia trascorso un tempo T, a partire dalla prima prenotazione per una determinata zona, il veicolo parte per la zona assegnata. Nel caso in cui non ci siano più posti disponibili su un veicolo, gli utenti in eccesso si mettono in attesa della prossima corsa per la zona desiderata.

**var:**

struct { *//usata per far si che ogni auto abbia i suoi semafori*

posti : semaforo generico (:=m) *//posti liberi sull’auto*

time : int (:=0) *//tempo trascorso*

contatore : int (:=0) *//contatore di persone salite sull’auto*

partenza : semaforo binario (:=0) *//semaforo usato per far partire l'auto*

prova\_partenza : semaforo binario (:=0) *//semaforo usato per provare a far partire l'auto*

mutex : semaforo binario (:=1) *//semaforo mutex per la mutua esclusione*

arrivo : semaforo binario (:=0) *//semaforo che indica l'arrivo*

pronto : semaforo binario (:=0) *//semaforo che serve per indicare che la macchina è disponibile.*

*Senza questo semaforo nel caso la macchina partirebbe per causa TEMPO*

*i clienti prenoterebbero lo stesso i posti.*

} auto;

auto car[N];

**utente (zona)**

wait (car[zona].pronto) *//attende che sia disponibile un auto in generale*

wait (car[zona].posti) *//attende che sia disponibile un auto con posti liberi*

wait (car[zona].mutex) *//modifico il contatore, cioè il numero di persone salite sull’auto*

car[zona].contatore = (car[zona].contatore + 1) mod M;

signal (car[zona].prova\_partenza)

signal (car[zona].mutex)

wait (car[zona].partenza)

VIAGGIO

wait (car[zona].arrivo)

LASCIA AUTO

**automobile (zona)**

repeat

repeat

signal (car[zona].pronto)

wait (car[zona].prova\_partenza)

wait (car[zona].mutex)

if (car[zona].contatore == 1) then

start timer

end if

posti\_occupati = car[zona].contatore

signal (car[zona].mutex)

until (posti\_occupati < M && car[zona].time < T)

signal (car[zona].partenza)

FAI LA CORSA

signal (car[zona].arrivo)

TORNA AL PUNTO DI PARTENZA

for (i=1 a i<=posti\_occupati)

signal(car[zona].posti) *//segnala la disponibilità di posti liberi sull’auto*

forever