1 Semafori

Lo scopo di questo esercizio è di farvi implementare semafori in C. Scrivete in un header file `my_semaphore.h` la struttura di dati e le prototipe come indicato sotto:

```c
#include <pthread.h>

typedef struct my_semaphore{
  volatile unsigned int V;
  pthread_mutex_t lock;
  pthread_cond_t varcond;
} my_semaphore;

int my_sem_init(my_semaphore *ms, unsigned int v);

int my_sem_wait(my_semaphore *ms);

int my_sem_signal(my_semaphore *ms);

int my_sem_destroy(my_semaphore *ms);
```

Vi chiediamo di implementare in un file `my_semaphore.c` le funzione `my_sem_init`, `my_sem_wait`, `my_sem_signal` e `my_sem_destroy` e di testarli in un programma che metterete in un altro file, per esempio implementando la soluzione dei filosofi con un salla d'attesa.

Per la semantica delle due ultime funzione `my_sem_init` e `my_sem_wait`è quella dei (weak) semafori presentati a lezione. Per quanto riguarda `my_sem_init`, inizializza i vari campi di un semaforo e `my_sem_destroy` libera i campi (chiamando le funzione `pthread_mutex_destroy` e `pthread_cond_destroy`). Ciascuna di queste tre funzione ritorna 0, se tutto è andato a buon fine,

2 Problema del bus

Abbiamo N passeggeri e un bus turistico con C posti (con $C < N$). Il comportamento del bus segue le regole:

(a) Aspetta di essere pieno (che C passeggeri sono saliti).

(b) Fa il giro della città.

(c) Arrive al punto di partenza e aspetta che tutti i passeggeri saliti scendono e torna in (a).

Il comportamento di un passagere segue le regole:

(a') Prova a salire nel bus se c'è ancora posta, altrimenti aspetta (in un modo passivo).

(b') Una volta salito, fa il giro turistico nel bus.

(c') Scende dal bus e torna in (a').

Usando dei lock, delle variabile condizionale o dei semafori, programmate questo sistemo dove il bus e ciascun passeggero verrà ripresentato da un thread. Ovviamente per testare vostra implementazione, dovrete fissare valori per C e N.

Indicazioni: Potete usare un contatore condiviso per contarre il numero di persone salite nel bus e dovete chiedervi chi deve 'svegliare' le entità in attesa (ad esempio l'ultimo passeggero a salire nel bus sveglierà il bus in attesa).

3 Consegna

Per la consegna, creare uno `zip` con tutti vostri file. Lo `zip` dovrà anche contenere un file `participanti.txt` dove gli nomi di chi ha participato alla consegna (questo anche se siete da solo a farla).