Σειρά Εργασιών 2

2.1 Δυαδικοί σηματοφόροι

Υλοποιήστε μια δική σας «βιβλιοθήκη» που παρέχει δυαδικούς σηματοφόρους με τις λειτουργίες:

`int mysem_init(mysem_t *s, int n);`	Αρχικοποίηση σηματοφόρου με τιμή n. Επιστρέφει 1 για επιτυχία, 0 αν n ≠ 0,1, και -1 αν ο σηματοφόρος είναι ήδη αρχικοποιημένος.
`int mysem_down(mysem_t *s);`	Μείωση σηματοφόρου κατά 1. Επιστρέφει 1 για επιτυχία ή -1 αν ο σηματοφόρος δεν έχει αρχικοποιηθεί.
`int mysem_up(mysem_t *s);`	Αύξηση σηματοφόρου κατά 1. Επιστρέφει 1 για επιτυχία, 0 αν ο σηματοφόρος είναι ήδη 1 ή -1 αν ο σηματοφόρος δεν έχει αρχικοποιηθεί.
`int mysem_destroy(mysem_t *s);`	Καταστρέφει τον σηματοφόρο. Επιστρέφει 1 για επιτυχία ή -1 αν ο σηματοφόρος δεν έχει αρχικοποιηθεί / έχει ήδη καταστραφεί.

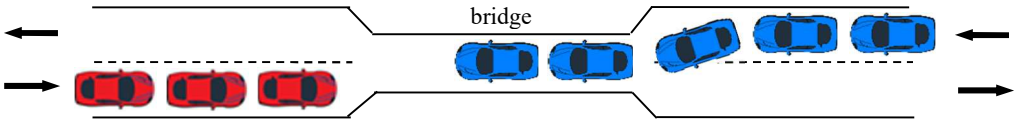
Η υλοποίησή σας πρέπει (εσωτερικά) να χρησιμοποιεί τους σηματοφόρους του system V (υποθέστε ότι είναι δίκαιοι). Για απλό αμοιβαίο αποκλεισμό η υλοποίηση σας μπορεί να χρησιμοποιεί συνδυαστικά και mutexes.

2.2 Αναγνώριση πρώτων αριθμών

Αλλάξτε τον κώδικα αναπτύξατε για την εργασία 1.2 έτσι ώστε όλος ο επιθυμητός συγχρονισμός (που υλοποιήσατε με ενεργή αναμονή πάνω σε κοινές μεταβλητές) να υλοποιείται τώρα με τους δικούς σας σηματοφόρους.

2.3 Στενή γέφυρα

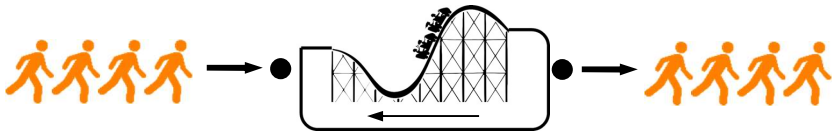
Αναπτύξτε κώδικα ελέγχου για τη ρύθμιση της κυκλοφορίας οχημάτων πάνω από μια γέφυρα, έτσι ώστε να διασφαλίζονται οι εξής ιδιότητες: (1) Δεν επιτρέπεται να υπάρχουν στη γέφυρα οχήματα που κινούνται προς αντίθετες κατευθύνσεις. (2) Δεν επιτρέπεται να υπάρχουν στη γέφυρα περισσότερα από N οχήματα.(3) Δεν επιτρέπεται ένα όχημα να περιμένει για πάντα να περάσει τη γέφυρα, ακόμα και αν καταφθάνουν συνεχώς οχήματα στην άλλη πλευρά.



Υλοποιήστε τον συγχρονισμό ανάμεσα στα οχήματα/νήματα, μέσω κατάλληλου κώδικα «εισόδου»/«εξόδου» που εκτελεί κάθε όχημα όταν φτάνει και αφού περάσει τη γέφυρα, αντίστοιχα. Ο χρόνος για το πέρασμα της γέφυρας προσομοιώνεται μέσω τεχνητής αναμονής. Ο συγχρονισμός πρέπει να γίνεται χωρίς κάποιο νήμα που να λειτουργεί ως μεσάζοντας/τροχονόμος. Η υλοποίησή σας πρέπει να βασίζεται στους δικούς σας δυαδικούς σηματοφόρους. Δοκιμάστε/επιδείξτε τη λύση σας μέσω ενός απλού προγράμματος προσομοίωσης που δημιουργεί οχήματα σε κάθε πλευρά της γέφυρας σε συγκεκριμένες χρονικές στιγμές με βάση πληροφορία που διαβάζει από την είσοδό του.

2.4 Τρενάκι

Το τρενάκι ενός λούνα-παρκ χωράει N επιβάτες. Το τρενάκι αρχίζει την επόμενη διαδρομή του μόνο όταν γεμίσει, ενώ οι επιβάτες αποβιβάζονται από το τρενάκι αφού αυτό ολοκληρώσει την τρέχουσα διαδρομή, και προτού αρχίσει η επιβίβαση των επόμενων επιβατών.



Υλοποιήστε τον επιθυμητό συγχρονισμό ανάμεσα στους επιβάτες και το τρενάκι, χωρίς κάποιο άλλο βοηθητικό νήμα. Η υλοποίησή σας πρέπει να βασίζεται στους δικούς σας δυαδικούς σηματοφόρους. Δοκιμάστε/επιδείξτε τη λύση σας μέσω ενός απλού προγράμματος προσομοίωσης που αρχικά δημιουργεί ένα νήμα για το τρενάκι και στην συνέχεια επιπλέον νήματα-επιβάτες σε συγκεκριμένες χρονικές στιγμές με βάση πληροφορία που διαβάζει από την είσοδο του. Ο αριθμός επιβατών που μπορεί να χωρέσει το τρενάκι δίνεται ως όρισμα του προγράμματος. Ο χρόνος που χρειάζεται το τρενάκι να κάνει την διαδρομή προσομοιώνεται μέσω τεχνητής αναμονής.

Η υλοποίηση πρέπει να γίνει σε C με χρήση της βιβλιοθήκης pthreads. Ο συγχρονισμός στις εργασίες 2.2-2.4 πρέπει να υλοποιηθεί αποκλειστικά με τους **δικούς σας δυαδικούς σηματοφόρους** και **χωρίς να υπάρχει ενεργή αναμονή**.

Παράδοση: Σάββατο 9 Νοεμβρίου 2024, 23:59