COGNOME..... NOME..... MATR. .....

SISTEMI OPERATIVI – Prova scritta - 7 settembre 2004

A – Un parco divertimenti ha installato una nuova attrazione “Cinema Vision 4D”. La sala di proiezione ha un numero limitato di posti NMAX=20. Per poter accedere allo spettacolo lo spettatore deve mettersi in coda fuori dell’edificio. All’ingresso dell’edificio è posizionato un tornello contapersone che può essere programmato, e riprogrammato, in modo da bloccarsi quando sono entrati spettatori nel numero programmato; è anche possibile in qualsiasi momento bloccare il tornello, portando a 0 il numero programmato. La proiezione, che dura TEMPOPROIEZIONE=5 minuti, ha di norma inizio non appena la sala si è riempita; volendo però garantire un’attesa massima degli spettatori in sala di ATTESAMAX=15 minuti, quando il primo spettatore entrato ha già atteso tale periodo di tempo, il tornello d’ingresso viene bloccato e la proiezione ha comunque inizio (quindi anche con un numero di spettatori inferiore a NMAX). Al termine della proiezione gli spettatori presenti in sala vengono fatti uscire attraverso un apposito tornello; svuotata la sala, il tornello d'ingresso viene riprogrammato.

1. Si descriva con una Rete di Petri estesa i vincoli e le sincronizzazioni che discendono dalle specifiche fornite, tenendo presente le modalità implementative (vedi punto successivo). In particolare si utilizzino posti esterni di ingresso per rappresentare l'arrivo degli spettatori e posti esterni di uscita per l’uscita degli spettatori dalla sala di proiezione.
2. Si realizzi, utilizzando i semafori (vedi nota *), il sistema sopra descritto. In particolare si realizzi la classe **`CV4D`** che, tra l'altro, include:
    a. Le dichiarazioni dei seguenti semafori:
        i. **`t1`** che rappresenta il tornello d'ingresso;
        ii. **`t2`** che rappresenta il tornello d'uscita;
        iii. **`postiOccupat`**i che rappresenta il numero di spettatori in sala (usato in particolare anche per segnalare l'ingresso in sala del primo spettatore del gruppo)
        iv. **`ultimo`** che serve per segnalare sua l'ingresso che l'uscita dell'ultimo spettatore del gruppo
    b. La classe interna **`Operatore`** che rappresenta il thread di controllo del sistema; il thread ciclicamente deve:
        i. Programmare il tornello di ingresso a NMAX;
        ii. Attendere l'ingresso del primo;
        iii. Dall'istante d'ingresso del primo attendere l'ingresso dell'ultimo: se viene superato il timeout ATTESAMAX, bloccare il tornello di ingresso;
        iv. Effettuare la proiezione;
        v. Far uscire dal tornello di uscita gli spettatori presenti in sala;
        vi. Attendere l'uscita dell'ultimo.
    c. La classe interna **`Spettatore`** che rappresenta, come thread, il singolo spettatore; esso deve:
        i. Attendere in coda davanti al tornello d'ingresso;
        ii. Occupare un posto in sala;
        iii. Se è l'ultimo del gruppo ad entrare, segnalarlo all'operatore di controllo;
        iv. Assistere alla proiezione;
        v. Prepararsi ad uscire sul tornello d'uscita.
        vi. Se è l'ultimo del gruppo ad uscire, segnalarlo all'operatore di controllo;
    d. Un metodo main di collaudo che crea una istanza dell'attrazione, un thread di controllo e un numero arbitrario di spettatori, generati in istanti casuali.

B – Uno stretto permette il collegamento fra il mare A ed il mare B. La larghezza del tratto navigabile permette il passaggio al massimo di due imbarcazioni da diporto. Per ragioni di sicurezza le imbarcazioni devono rimanere in attesa per il minor tempo possibile da entrambi i lati, ossia se da un lato vi sono una o due barche, e una sola dall’altro lato, viene obbligato il transito nei due versi. Se da entrambe le parti le imbarcazioni sono più d’una, viene autorizzato il transito a senso unico alternato di coppie di imbarcazioni fino a che non ci si riporta in una situazione in cui o c'è al più una barca in attesa oppure è ammesso il transito nei due versi.

Formalizzare in ADA il codice esemplificativo del task di controllo dello stretto e dei task che rappresentano le singole imbarcazioni.

Il task **`Imbarcazione`** utilizza uno degli entry **`entraAB`**, **`esceAB`**, **`entraBA`**, **`esceBA`**, dove i suffissi AB e BA rappresentano il rispettivo verso di attraversamento. La realizzazione dovrà provvedere le sincronizzazioni previste dalle specifiche nonché la totalizzazione dei transiti nei rispettivi versi delle imbarcazioni.

(Si ricorda che la primitiva **`E'COUNT`** restituisce il numero di task sospesi in quell'istante sull'entry **`E`**.)

(*) se serve, si assuma di disporre del package Os,
da non trascrivere in alcuna sua parte nella soluzione fornita,
limitandosi ad importare da esso le classi utilizzate.