COGNOME _____ NOME_____ MATR. _____

SISTEMI OPERATIVI – Esempio Prova scritta

A – Si vuole simulare l'attività di carico di una nave mercantile attraccata al fianco a un molo portuale. Le merci sono contenute in container che arrivano presso la nave su camion (un container per ciascun camion). Sul molo (affiancate alla nave) ci sono 2 gru (G1, G2), che corrono su un unico binario lungo tutta la fiancata della nave. Sono previsti 3 distinti punti di carico: il punto A, riservato per le merci deperibili, e i punti B e C per il carico delle altre merci. Le merci arrivano su camion che si posizionano singolarmente nei punti di carico (quando liberi) e attendono che una gru carichi sulla nave il container che trasportano. Ogni camion, appena vuoto, abbandona il molo lasciando il posto al successivo. I camion con merci non deperibili non possono posizionarsi nel punto A. Per lo scarico, le gru si affiancano ai camion nei punti di carico correndo lungo l'unico binario. Ogni gru può spostarsi presso il punto di carico adiacente (se non è già presente un'altra gru). Se un camion è in attesa al punto A, la gru G1 si occupa di scaricare quel camion, con priorità mentre l'altra continuerà a scaricare gli altri eventuali camion in attesa nei punti B e C. Solo se il punto A risulta vuoto la gru G1 può scaricare i camion con merci non deperibili e posizionati nel punto B.
Utilizzando i Semafori come costrutto di sincronizzazione si realizzi la classe Molo che simula il sistema di controllo e comprende **tra l'altro**:

- Le variabili di conteggio **truckD**, **truckS** che rappresentano rispettivamente il numero di camion in attesa con merci deperibili o con merci standard (non deperibili)
- La variabile **puntoDiCarico[3]** che rappresenta, per ciascun posto di carico se libero
- Un semaforo di mutua esclusione **mutex**, e un vettore **att[2]** di semafori (privati) per le attese di accesso al servizio;
- Un vettore **stopGru[2]** di semafori per gestire l'attività delle Gru e un ulteriore array di semafori **servito[]** per notificare al camion che lo scarico è terminato
- La classe interna **Camion** che rappresenta un generico thread camion il cui tipo di merce e il cui indice sono forniti come parametro del costruttore **Camion(boolean, int)**. Nel suo metodo principale accede al molo attendendo che un punto di carico pertinente sia libero, utilizzando il metodo **int entra(boolean)**. Raggiunto il punto di carico attende che il posto sia attrezzato (ci sia una gru che puo` servirlo) e che lo scarico sia terminato (il termine dello scarico è segnalato dalla gru assegnata). Al termine lascia il molo liberando il posto, utilizzando il metodo **void esce(int)**.
- Il metodo **int entra(boolean richiestaA)** che verifica se un punto di carico è libero con una eventuale attesa su un semaforo privato. Dopo l'assegnazione del punto di carico, seleziona la gru da cui attendere il servizio e ritorna l'indice del punto di carico selezionato;
- Il metodo **void esce(int punto)**: che rappresenta la terminazione di uno scarico e la conseguente liberazione del posto di carico che è passato come parametro.
- La classe interna **Gru** per simulare i posizionamenti delle gru lungo il binario, Nel metodo principale una gru attende che ci sia una richiesta di servizio a lei indirizzata. Quando entrambe le richieste sono presenti la gru G1 sceglie sempre il punto A mentre per G2 la scelta è indifferente. La fornitura del servizio è simulata con un'attesa casuale compresa tra **tMin** e **tMax**, parametri passati in fase di costruzione dell'istanza della classe **Molo**.
- Il metodo **main** che, dopo aver inizializzato l'istanza della classe **Molo**, acquisendo i due parametri di costruzione dalla linea di comando, genera con tempi casuali uno dei due tipi di camion.

Il candidato non deve preoccuparsi di controllare che la massima capienza della nave sia stata raggiunta durante le operazioni di carico.

#

B – Si voglia simulare in **ADA-Java**, il controllo di un magazzino di ricambi, tutti dello stesso tipo. La fornitura di nuovi pezzi al magazzino avviene per lotti, e può essere di due tipi, A di quantità **npa** più rilevante con QM ≥ **npa** ≥ QP, B di quantità **npb** inferiore con **npb** < QP. Il prelievo di ricambi è invece unitario. Il magazzino ha una capacità massima MAXP (>QM). La simulazione prevede i seguenti *entry*:

- **prod**
  chiamato all'atto di una produzione di un lotto A o B, la quantità **np** viene passata come parametro. Per garantire il non superamento della capienza, l'entry è bloccante se il magazzino contiene già più di MAXP-QM pezzi di ricambio.
- **lottoA**
  chiamato subito dopo **prod** da un produttore di un lotto di tipo A: il metodo è bloccante se l'attuale disponibilità è ≥ QM/4 (serve per favorire la piccole produzioni di lotti B quando c'è basso consumo); fare in modo che l'attesa del thread produttore non si prolunghi oltre un tempo TMAX;
- **prel**
  chiamato da un thread consumatore per prelevare un ricambio.
  Fornire, utilizzando se lo si desidera la notazione con macro, il costrutto **`select`** del thread server di simulazione del controllo del magazzino.

N.B. Se si risolve correttamente solo il tema A il massimo punteggio ottenibile è 26/30; la risoluzione del solo tema B non fa raggiungere la sufficienza.