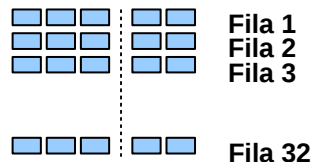


## Esame di Sistemi Operativi del 30 Luglio 2010 - Tempo a disposizione 3.5 ore

Si vuole progettare un sistema per la gestione dell'assegnamento dei posti sui voli di linea per gli aerei modello MD-80. Tali aerei mettono a disposizione 32 file di posti. Ogni fila è divisa dal corridoio in un gruppo di 2 posti (finestrino, corridoio) e di 3 posti (corridoio, centro, finestrino).



A tal proposito, si deve progettare la classe **GestorePrenotazioni**, la quale contiene come suoi campi:

**STRUTTURE DATI** (E' a carico dello studente stabilire quali e come le seguenti strutture dati debbano essere sincronizzate per garantire accesso in mutua esclusione)

1. un istanza della classe **Aereo** che chiameremo A: un Aereo tiene traccia dello stato corrente dell'occupazione dei posti e fornisce i metodi per assegnare ciascun posto ai passeggeri che abbiano un biglietto sull'aereo in questione. Ogni singolo posto può essere *libero* oppure assegnato a un determinato passeggero. Ogni passeggero è identificato da un codice univoco (un valore intero positivo).

In particolare la classe *Aereo* fornisce il metodo

*bool allocaPosto(int codPasseggero, Coord P).* Assegna il posto di coordinate P al passeggero con codice codPasseggero. La classe Coord (da progettare a carico dello studente) identifica le coordinate di un certo posto. Questo metodo è **l'unico dal quale si possano fare modifiche allo stato di assegnazione dei posti**. Perciò per allocare più di un posto è necessario invocare questo metodo per più volte.

E' a carico dello studente l'aggiunta di eventuali altri metodi addizionali.

2. un array di **BancoCheckin**, che chiameremo B: un banco checkin consiste in una struttura dati che accetta una serie di comitive *nell'ordine in cui si presentano al banco*. Ogni comitiva è composta da 1 o più viaggiatori. Metodi da implementare:

*Coord\* riservaPosti(int\* codPasseggeri, int L).* Trova posto sull'aereo A alla comitiva costituita da L passeggeri il cui rispettivo codice è indicato nell'array codPasseggeri, di L elementi. Restituisce la "carta d'imbarco", costituita da un array di Coord, il quale contiene i posti assegnati alla comitiva. Il metodo ritorna *null* se non c'è posto per tutti i membri della comitiva. Questo metodo: a) inserisce una richiesta di allocazione posti nel banco checkin; b) attende che questa sia servita da un operatore di banco; c) ritorna la carta di imbarco.

**THREAD** (E' a carico dello studente stabilire quali e quante istanze di ciascun tipo di thread debbano essere avviate e in quale momento)

**GeneratoreComitive.** Il GestorePrenotazioni ne possiede uno, che chiameremo G. Il generatore G genera e avvia continuamente delle comitive di una certa dimensione (da 1 a 5).

**Comitiva.** Una volta generata, una comitiva sceglie un determinato banco checkin tra quelli presenti nell'array B, invoca opportunamente il metodo riservaPosti, ed esce dopo aver stampato la carta d'imbarco a video.

**OperatoreBanco.** Per ogni bancoCheckin presente in B, esiste un corrispondente operatore al banco. L'operatore compie il seguente ciclo continuo: 1) preleva una richiesta di riserva di posti dal proprio banco; 2) **Dopo avere ottenuto accesso esclusivo all'aereo A**, cerca di allocare un posto per ciascun membro della comitiva **sulla stessa fila** (se possibile), invocando a più riprese il metodo *allocaPosto* di A. Un operatore termina nel momento in cui l'aereo A è *completamente pieno*.

*E' parte integrante della prova di esame*

- a) *completare le specifiche date nei punti non esplicitamente definiti;*
- b) *introdurre eventuali campi e metodi ausiliari che si ritengano necessarie;*
- c) *risolvere eventuali ambiguità;*
- d) *garantire accesso in mutua esclusione alle strutture dati ove si ritenga necessario;*
- e) *evitare possibili situazioni di starvation o di deadlock.*

**Non è consentito modificare il prototipo dei metodi di classe se questo è stato fornito.**

