Sistemi Operativi

Compito 22-06-2021

Esercizio 1 (10 punti)

In un sistema sono presenti tre processi P_1, P_2, P_3 che usano quattro tipi di risorsa A, B, C, D di cui sono presenti 4 unità per ciascun tipo. I processi hanno le seguenti matrici di allocazione delle risorse e di massima richiesta di risorse:

	Allocazione				Max			
	A	**B**	**C**	**D**	**A**	**B**	**C**	**D**
P_1	2	1	2	0	4	1	2	1
P_2	1	0	1	2	1	1	1	3
P_3	1	1	1	1	2	2	2	2

Usando l'**algoritmo del banchiere** stabilire se il sistema si trova in uno stato sicuro ed in caso positivo indicare tutte le possibili sequenze sicure. In un istante successivo arriva il processo P_4 che usa al massimo una risorsa per tipo, dire se questo nuovo stato è sicuro o meno ed elencare le possibili sequenze sicure. Per tutte le possibili richieste valide di risorse disponibili e compatibili con la richiesta massima del processo P_4 determinare se portano in uno stato sicuro o meno.

Esercizio 2 (20 punti)

Si vuole simulare il seguente sistema:

Sono presenti N veicoli che si muovono in città ed acquisiscono una immagine al secondo, ogni veicolo invia la sua posizione (x,y) ad un *LocationTracker* che risponde con indicazione se l'immagine è ad alta/media/bassa priorità. Le immagini vengono inserite in una coda locale al veicolo sulla base della loro priorità. Per ogni veicolo un thread *Uploader* preleva le immagini dalla coda locale sulla base della loro priorità e le inserisce in una unica coda di dimensione massima K dalla quale M *ImageCollector* le estraggono impiegando un tempo T1.

Per il testing posizionare i veicoli in modo casuale nel quadrato x in [-10,10) y in [-10,10), aggiornare la posizione del veicolo ogni secondo con dx casuale in [-1,1) e dy casuale in [-1,1). Dove x, y, dx e dy sono numeri in floating point. Il *LocationTracker* deve considerare ad alta priorità immagini di veicoli che si trovano a meno di 5 dal punto 0,0 e a media priorità quelli che distano tra 5 e 10 e a bassa priorità i restanti, inoltre deve contare il numero di veicoli che hanno comunicato la loro posizione nell'ultimo secondo divisi per priorità (quando il contatori vengono acquisiti i contatori vanno azzerati evitando che si abbiano perdite e race conditions).

Il programma principale deve inizializzare i thread farli partire e ogni secondo stampare il numero di veicoli che nell'ultimo secondo hanno comunicato di trovarsi in zona ad alta, media e bassa priorità e quante immagini sono presenti nella coda di upload divise per priorità. Dopo 30 secondi terminare tutti i thread e stampare la somma del numero di aggiornamenti di posizione fatti da ciascun veicolo e il totale dei cambiamenti di posizione acquisiti durante i 30 secondi (questi due numeri dovranno essere uguali).

Realizzare in java il sistema descritto usando i ***metodi sincronizzati*** per la sincronizzazione tra thread.