Sistemi Operativi

Compito del 19-06-2024

Esercizio 1 (10 punti)

In un sistema sono presenti quattro processi P_1, P_2, P_3 e P_4 che usano quattro tipi di risorsa A, B, C, D di cui sono presenti 3 unità per ciascun tipo. I processi hanno la seguente matrice di allocazione delle risorse e di uso massimo:

	Allocazione				Max			
	A	**B**	**C**	**D**	**A**	**B**	**C**	**D**
P_1	1	2	1	0	2	2	2	0
P_2	0	1	0	1	0	1	0	2
P_3	0	0	0	1	1	0	0	1
P_4	2	0	1	0	2	1	1	0

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. Quindi indicare per ogni processo se in questo stato le possibili richieste di risorse disponibili portano il sistema in uno stato sicuro o meno (giustificare il risultato).

Esercizio 2 (20 punti)

Si vuole realizzare il seguente sistema:

sono presenti N *ProcessorThread* dove iterativamente ognuno: chiede un array di K elementi ad un oggetto condiviso *ArrayGenerator*, acquisisce una o due risorse A da un *ResourceManager*, produce un risultato impiegando un tempo T, rilascia le risorse acquisite e inserisce il risultato insieme all'array in una coda limitata di lunghezza L.

L'*ArrayGenerator* può generare un solo array alla volta (in caso di richieste concorrenti solo uno riesce ad acquisire, gli altri aspettano) ed impiega un tempo TG.

Il *ResourceManager* gestisce NA risorse A e chi chiede aspetta se le risorse richieste non sono tutte disponibili.

Sono presenti due *OutputThread* dove ognuno preleva dalla coda 2 risultati e li stampa.

Per facilitare il testing *l'ArrayGenerator* genera array di numeri in sequenza [1,2,..K], [2,3,..K+1], [3,4,…,K+2] e i *ProcessorThread* calcolano la somma dei valori nell'array.

Il programma principale deve far partire i thread necessari e dopo 10 secondi terminare tutti i thread e stampare: quanto array sono stati generati, il numero di risorse disponibili, per ogni ProcessorThread quanti array ha processato, quanti risultati sono in coda e per ogni OutputThread il numero di stampe fatte.

Inoltre si faccia particolare attenzione a rilasciare sempre le risorse acquisite.

Realizzare in java il sistema descritto usando i ***semafori*** per la sincronizzazione tra thread (sarà considerato errore usare polling o attesa attiva).