Sistemi Operativi

Prima prova in itinere 26-04-2017 - **Compito B**

Esercizio 1 (12 punti)

Un sistema operativo adotta la politica di scheduling dei thread a code multiple con *prelazione* tra le code. Sono presenti tre code, una a priorità 1 (alta priorità) con scheduling round-robin con quanto q=2ms, una coda a priorità 2 (media priorità) con scheduling round-robin con quanto q=2ms e una coda a priorità 3 (bassa priorità) con scheduling SJF (senza prelazione all'interno della coda).

Il sistema deve schedulare i seguenti thread con tempi di arrivo, priorità e uso CPU/IO indicati:

T_1 T_{arrivo}=1 pri=1 CPU(1ms)/IO(3ms)/CPU(3ms)
T_2 T_{arrivo}=1 pri=2 CPU(1ms)/IO(3ms)/CPU(1ms)
T_3 T_{arrivo}=0 pri=3 CPU(2ms)/IO(1ms)/CPU(3ms)
T_4 T_{arrivo}=0 pri=3 CPU(3ms)/IO(1ms)/CPU(2ms)

Si determini:

1. il **diagramma di Gantt**,
2. il **tempo di attesa** medio,
3. il **tempo di ritorno** medio,
4. il **tempo di risposta** medio,
5. il numero di cambi di contesto e
6. gli istanti in cui si ha prelazione tra le code.

Esercizio 2 (18 punti)

In un sistema sono presenti N thread *MakeArrayThread* che generano ognuno un array di M numeri interi generati in modo randomico con valore compreso tra 1 e 100 (estremi compresi). Il thread *CountEvenThread* prende gli array generati e genera un array di M valori interi dove per ogni posizione indica il numero di valori pari degli N array in quella posizione e lo stampa. Inoltre gli N thread *MakeArrayThread* ed il thread *CountEvenThread* sono fatti partire in modo periodico ogni 3 secondi. Implementare in Java quanto descritto usando il metodo join per sincronizzare i thread e chiedendo i numeri N e M in input da tastiera.