L'obbiettivo di questa serie d'esercizi è d'imparare a riconoscere e correggere i problemi di visibilità della memoria. Inoltre, l'obbiettivo è quello d'imparare ad utilizzare la keyword "*volatile*" e le atomic variables di Java.

Esercizio 1

Copiate il codice sorgente 'S3Esercizio1.java' e analizzatelo nel dettaglio. Provate a compilarlo ed eseguirlo. Cercate di capire la natura del problema di cui soffre. In seguito, provate a risolvere il problema.

Esercizio 2

Scrivete un programma che simuli il comportamento di 10 sensori per il monitoraggio di un contatore condiviso. Ogni sensore dovrà avvisare, qualora il valore del contatore dovesse oltrepassare una certa soglia. Ogni sensore verrà calibrato con una soglia differente: 10 per il primo, 20 per il secondo fino a 100 per l'ultimo. Nell'istante in cui un sensore dovesse accorgersi che il contatore supera il valore soglia, dovrà notificare il superamento della soglia, azzerare il contatore e terminare la propria esecuzione. Il programma principale si occuperà di creare i 10 sensori e di farli partire. In seguito, il programma principale dovrà iniziare ad incrementare con un incremento casuale tra 1 e 8 la variabile condivisa, aspettando un tempo casuale tra 5 e 10 ms fra un incremento e l'altro. Il programma dovrà completare qualora il valore del contatore dovesse oltrepassasse 120.

Provate ad implementare una prima versione senza utilizzare strumenti di sincronizzazione. Che problemi ci sono? Successivamente, provate ad implementarne una versione che utilizzi una variabile volatile. Riuscite a risolvere i problemi di sincronizzazione? In alternativa, verificate se è possibile risolvere il problema utilizzando le variabili atomiche. Infine, sviluppate due ulteriori versioni; la prima che utilizza gli explicit lock, la seconda i ReadWriteLock. Ci sono vantaggi? Se si, quali?

Esercizio 3

Scrivete un programma composto da 10 worker thread e da un array condiviso di 5 interi inizializzato a 0. Ogni thread dovrà eseguire 10'000 volte le seguenti operazioni:

- scegliere una posizione a caso dell'array
- sommare al valore della posizione scelta un numero casuale tra 10 e 50
- se il nuovo valore supera 500 rimettere il valore della posizione a 0
- prima di scegliere la prossima posizione dovrà attendere un tempo casuale tra 2 e 5 ms

La protezione dello stato condiviso deve avvenire esclusivamente mediante lock espliciti. Assicuratevi che il programma termini correttamente attendendo che tutti i threads abbiano completato.