Specifiche progetto Sistemi Operativi sessione estiva A.A. 2011/2012
Docente Emanuele Lattanzi

1. Specifiche generali

Scrivere un programma multithread che simuli il traffico automobilistico. In particolare il programma dovrà simulare il traffico automobilistico su di una strada a due corsie e doppio senso di marcia, che procede in direzione Nord-Sud. Lungo tale strada vi è un ponte ad una sola corsia, percorribile a senso unico alternato. Il ponte ha la capacità di *n* automobili. Le automobili dirette verso Nord (oppure verso Sud) possono attraversare il ponte solo se non è attraversato da nessuna automobile diretta verso Sud (oppure verso Nord). Il programma dovrà garantire l'assenza di incidenti automobilistici (*race condition*) e dovrà permettere di valutare i tempi d'attesa per salire sul ponte delle auto dirette verso Sud e delle auto dirette verso Nord.
Il programma dovrà essere costituito da un oggetto "ponte" (PONTE) che sarà un oggetto condiviso tra un numero arbitrario di thread (*m*) che rappresentano le automobili (AUTO).
Il programma terminerà quando ogni automobile sarà riuscita ad attraversare il ponte.

2. Politica di gestione delle auto in attesa

La politica di gestione delle attese è una semplice politica di tipo FIFO. Cioè le auto saliranno sul ponte secondo l'ordine di arrivo (per prime le auto che sono arrivate per prime a ridosso del ponte.

3. Specifiche dell'oggetto PONTE

- L'oggetto PONTE sarà un oggetto condiviso tra gli *(m)* thread che rappresentano le automobili.
- L'oggetto PONTE dovrà emulare un ponte a singola corsia a senso unico alternato percorribile al più da *(n)* automobili alla volta.
- Il PONTE sarà caratterizzato da un tempo di attraversamento pari a 100ms.
- L'oggetto PONTE esporrà due metodi pubblici *entra(...)* ed *esci(...)* che saranno utilizzati dalle auto rispettivamente per richiedere di entrare sul ponte e per notificarne l'uscita.
- Il metodo *entra()* dovrà essere bloccante ….

4. Specifiche dell'oggetto AUTO:

- L'oggetto AUTO dovrà emulare un'automobile che, arrivando a ridosso del ponte ne richiederà l'attraversamento utilizzando il metodo pubblico dell'oggetto PONTE (*entra(...)*) descritto in precedenza. Una volta ottenuto il consenso ad attraversare il ponte l'automobile dovrà simularne l'attraversamento sospendendosi per il tempo necessario.
- Ogni automobile alla fine dell'attraversamento dovrà utilizzare il metodo (*esci(...)*) dell'oggetto ponte per notificare alle altre auto la sua uscita dal ponte.
- Ogni AUTO dovrà essere emulata attraverso l'utilizzo di un singolo thread.
- Ogni AUTO dovrà essere caratterizzata da una direzione di marcia (Nord-Sud o Sud-Nord) definita casualmente all'atto della creazione dell'oggetto AUTO.
- Il comportamento di ogni oggetto AUTO può essere descritto dai seguenti passi:
    1. Attendi un tempo casuale compreso fra 10 e 20 ms per simulare l'arrivo casuale a ridosso del ponte.
    2. Effettua la richiesta di entrare sul ponte.
    3. Appena viene concesso l'attraversamento simula il tempo necessario
    4. Notifica alla altre auto la fine dell'attraversamento
    5. Esci.

5. Specifiche di sincronizzazione ed implementazione:

- Il progetto deve essere sviluppato in ambiente Linux/Windows utilizzando **esclusivamente il linguaggio java.**
- La muta esclusione, l'accesso a variabili e a metodi condivisi e l'attesa dei thread dovranno essere ottenute utilizzando le primitive di sincronizzazione messe a disposizione dal linguaggio java (ad esempio: package java.util.concurrent.) ad esclusione del costrutto monitor (per chiarezza non sarà possibile utilizzare il costrutto *synchronized*) .

6. Specifiche dati in input:

- Il programma dovrà ricevere in input, tramite riga di comando o tramite file di specifica, i seguenti parametri:
- Numero massimo di automobili che possono percorrere il ponte contemporaneamente *(n)*.
- Numero di automobili *(m)*.

7. Specifiche dati di output:

- Al termine della simulazione il programma dovrà scrivere, su di un file di testo oppure sulla shell, il tempo d'attesa per ogni automobile ed il tempo medio d'attesa per le auto che viaggiano in direzione Nord e per le auto che viaggiano in direzione Sud (*tempoAttesa* - tempo che intercorre tra la richiesta di attraversare il ponte e l'inizio dell'attraversamento). Per rilevare i tempi d'attesa è consigliato l'utilizzo del metodo currentTimeMillis() della classe System.

8. Documentazione

- Il progetto deve essere corredato dalla seguente documentazione:
  - codice sorgente commentato nei punti salienti
  - eventuali file di configurazione dell'applicativo
  - relazione in formato PDF o Word contenente:
    - specifiche richieste
    - schema a blocchi (sufficientemente dettagliato) degli elementi progettuali del codice: classi, oggetti e loro connessione logica
    - descrizione della progettazione
    - descrizione dell'implementazione
    - testing e commenti ai risultati ottenuti.

- La relazione, in formato PDF, deve essere inserita in una directory (denominata docs), da allegare al progetto stesso. Vedere il seguente link per una relazione d'esempio: http://www.sti.uniurb.it/lattanzi/OS/Relazione_SO.pdf

9. Modalità di presentazione del progetto:

- Il presente progetto può essere presentato fino ad UNA SETTIMMANA prima dell'appello d'esame in cui s'intende sostenere la prova orale.

- La consegna del progetto dovrà avvenire esclusivamente per posta elettronica all'indirizzo emanuele.lattanzi@uniurb.it avendo l'accortezza di inserire nel campo subject i seguenti dati:

  Progetto SO/OSys - <numero matricola> - Nome_e_cognome.

  SO: per il corso di sistemi operativi percorso in presenza
  OSys: per il corso di Operatine System percorso on-line

- Per ulteriori informazioni su data e ora di consegna del presente progetto consultare il sito personale del docente al link: http://www.sti.uniurb.it/lattanzi/OS.html
- Il progetto deve pervenire, come allegato, sotto forma di file compresso in modalità .tar.gz o zip.
- Progetti consegnati fuori tempo limite non verranno accettati.
- Progetti consegnati senza relazione oppure accompagnati da relazione che non soddisfi le specifiche richieste non saranno ritenuti sufficienti.

10. Divieto di plagio:

- Ogni progetto, elaborato autonomamente e personalmente da ciascuno studente, deve essere originale. Codice, o porzione di codice che verrà identificato come "copiato" da altre fonti comporterà l'automatica valutazione insufficiente del progetto. Fanno doverosamente eccezione a questa norma i riferimenti a packages e librerie reperite nei repository pubblici per i quali sarà comunque necessario citarne la fonte d'origine in un'apposita sezione bibliografica della relazione.
- Progetti elaborati da persona o persone diverse dallo studente che li sottomette saranno automaticamente valutati insufficienti.

11. Avvertenze generali:

- Le presenti specifiche sono valide a decorrere dalla loro data di pubblicazione fino alla fine della sessione d'esame esplicitata nell'intestazione del presente documento.
- La presente specifica rende nulla qualsiasi altra specifica pubblicata in data antecedente, e qualsiasi informazione, relativa alla stesura, implementazione, consegna e valutazione del progetto, reperita da qualsivoglia altra fonte ufficiale e non.