Nome: Cognome: Compito: Turno:

PRINCIPI DI SISTEMI OPERATIVI
(A.A. 11-12)
20 Giugno 2012

IMPORTANTE:

1. Si considerano parte integrante delle soluzioni i **COMMENTI *significativi*** introdotti per facilitare la lettura del codice: come tali, essi influenzano la votazione finale. Tuttavia, i messaggi di debug (ad es. le println()) del programma NON SONO CONSIDERATI E QUINDI NON INFLUENZANO LA VOTAZIONE FINALE.
2. Il tempo a disposizione è di 90 minuti.
3. Il compito deve essere svolto solamente nel linguaggio Java, usando le classi del package **monitor** e lavorando con l'ambiente di sviluppo **IBM Eclipse**.
4. Seguire le seguenti regole per lo svolgimento dell'esame al laboratorio base:
   - Fare il login in Linux con il proprio account (numero di tesserino e password di posta elettronica)
   - Aprire un terminale e digitare
     ```
     $ cd
     $ cd Desktop
     $ wget ftp://lica.lab.unimo.it/syncexam.sh
     $ chmod 755 ./syncexam.sh
     $ ./syncexam.sh
     ```
   - Aprire Elipse (comando "eclipse" sempre da shell)
   - Utilizzare come workspace la cartella "studente_xxxxx"
   - Creare un progetto Java con nome "ESAME200612-<<turno>>-<<n>>" e scrivere le classi Java della soluzione nel package di default (senza nome) di tale progetto. Fare attenzione a scrivere correttamente il nome del progetto, con maiuscole e minuscole a posto!
   - Installare le classi del monitor Java e gli eventuali template
   - Finito il vostro esame (o allo scadere del tempo), dovete salvare tutto (si consiglia di salvare spesso per non perdere il proprio lavoro), chiudere Eclipse, fare il logout, lasciare il vostro PC e procedere alla consegna del testo.

In una grande **Città** si avverte una potente scossa di terremoto. Gli N **cittadini** che abitano in quella città, una volta passata la paura, contattano o i **VVFF** o la **Protezione Civile** a seconda della classificazione della zona in cui abitano (zona rossa, gialla o verde scelta in maniera random alla creazione del processo).
I cittadini che abitano nella zona rossa contattano i VVFF che devono mandare una delle loro S squadre (S << N) a verificare l'agilità dell'abitazione. La prima squadra dei VVFF libera, si reca da un cittadino, prendendo la sua chiamata in ordine di arrivo. Se la casa viene dichiarata dai VVFF "inagibile" (per simulare tale decisione si usi una scelta random), il cittadino deve chiedere aiuto alla Protezione Civile (v. dopo), mentre se viene dichiarata "agibile" il cittadino può tornarsene a casa.
I cittadini che abitano nella zona gialla contattano la Protezione Civile, per ottenere un controllo strutturale dell'abitazione, mentre i cittadini che abitano nella zona rossa la cui casa è stata dichiarata "inagibile" chiedono, sempre alla Protezione Civile, ospitalità nelle tendopoli.
La Protezione Civile ha C squadre (C < N). Ogni squadra può effettuare una verifica strutturale per i cittadini della zona gialla (verifica che dura un tempo t scelto random), oppure può aprire le pratiche per l'ospitalità nella tendopoli per i cittadini della zona rossa (tempo per le pratiche z, scelto ancora in modo random). La Protezione Civile dà la priorità ai cittadini che abitano nella zona rossa.
I cittadini che invece abitano nella zona verde, devono attendere che sia finita l'emergenza (ovvero che non ci siano cittadini della zona rossa che chiedono l'intervento dei VVFF), per chiedere ai VVFF di aprire una pratica per verificare eventuali crepe (tempo per l'apertura della pratica scelto random).

Si implementi una soluzione usando il costrutto monitor per modellare la **Città**, i processi per modellare i **cittadini**, i **VVFF** e la **Protezione**. Nella soluzione si massimizzi l'utilizzo delle risorse. Si discuta se la soluzione proposta può presentare starvation e in caso positivo per quali processi, e si propongano modifiche e/o aggiunte per evitare la starvation.