Nome: Cognome: Compito: Turno:

PRINCIPI DI SISTEMI OPERATIVI
(A.A. 11-12)
18 Gennaio 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
   - Eseguire i seguenti comandi shell:
     a. *mkdir rhome* per creare una directory locale 'di appoggio'
     b. *sshfs USER@155.185.31.1XX:/homel/nUSER rhome* (dove USER è il numero di tesserino universitario e XX è il numero della macchina fisica LINFA) per montare la propria home directory remota che risiede sul server virtuale (lica.lab.unimo.it) sulla directory locale 'di appoggio'
        Questo comando richiederà per essere eseguito di inserire la password di posta elettronica
   - Aprire Elipse (comando "eclipse" sempre da shell) e scegliere come workspace nella propria home directory locale la cartella rhome sulla quale avete appena montato la directory remota (workspace = user/rhome)
   - Creare un progetto Java con nome "ESAME180112-2-12" 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 un **Centro per l'Agopuntura** lavorano D **dottori** (con D multiplo di 2). Nel Centro si recano i **pazienti** (bambini o adulti) per fare una seduta di agopuntura.
Una volta trovato un dottore in grado di iniziare una seduta (si veda nel seguito le specifiche condizioni), il paziente attende il termine del trattamento per un tempo random (scelto dal paziente stesso). Nelle sedute di agopuntura, i bambini hanno la priorità rispetto agli adulti. Una volta finita la seduta, il paziente libera il dottore e torna a casa.
Dei D dottori che lavorano nel centro, D/2 sono primari e D/2 dottori non primari. I primari sono in grado di effettuare una seduta:

a) se non hanno ancora effettuato N sedute di agopuntura;
b) hanno effettuato N sedute di agopuntura e si sono fermarti per un tempo X (noto) per sbrigare le pratiche burocratiche.

Quindi se un paziente trova un dottore primario libero che ha già effettuato N visite, deve anche verificare che abbia finito le pratiche per poter iniziare la seduta*; se un paziente non trova nessun dottore in grado di iniziare una seduta, il paziente deve attendere un dottore.
Si implementi una soluzione usando il costrutto monitor per modellare il **Centro per l'Agopuntura**, i processi per modellare i **pazienti** (adulti e bambini) e si utilizzino i **dottori** come risorse. 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.

---

* Suggerimento: Si memorizzi l'informazione relativa al tempo in cui un paziente libera un primario, ricavata usando il metodo getTime() della classe java.util.Date.