NON SPEGNERE IL PC A FINE ESAME

Corso di Sistemi Operativi e Reti

Corso di Sistemi Operativi

Prova scritta di NOVEMBRE 2017

ISTRUZIONI

1. **Rinomina** la cartella chiamata "CognomeNomeMatricola" che hai trovato sul Desktop e in cui hai trovato questa traccia, sostituendo "Cognome" "Nome" e "Matricola" con i tuoi dati personali;
2. **Carica** tutto il materiale didattico che vorrai usare sul Desktop; puoi farlo solo nei primi 5 minuti della prova;
3. **Svolgi** il compito; lascia tutto il sorgente che hai prodotto nella cartella di cui al punto 1;

4. Quando hai finito lascia la postazione facendo logout,

senza spegnere il PC.

SALVA SPESSO il tuo lavoro

PER GLI STUDENTI DI SISTEMI OPERATIVI: si può sostenere solo uno dei due esercizi se si è già superata in un appello precedente la corrispondente prova. Il tempo a disposizione in questo caso è di 2 ORE.

NON SPEGNERE IL PC A FINE ESAME

ESERCIZIO 1 (Programmazione multithread. Punti: 0-20)

Si implementi una classe `ReadWriteLockEvoluto` secondo le specifiche di seguito riportate.

La classe `ReadWriteLockEvoluto` si comporta come la classe `ReentrantReadWriteLock` di Java ma aggiunge i seguenti metodi:

`void setReaders(int max_readers).` Il metodo imposta un limite sul numero massimo di lock in lettura possibili sul `ReadWriteLockEvoluto` in esame e termina immediatamente. Ulteriori richieste di locking che successivamente eccedano il numero precedentemente impostato di readers devono essere poste in attesa. Nel caso sia già presente un numero di lock in lettura eccedente i max_readers, i lock già presenti possono essere invece mantenuti ma mai aumentati. Il limite di default per il numero massimo di lettori su un ReadWriteLockEvoluto è 10.

Esempi:

1. Si assuma che il numero massimo di lettori sia stato precedentemente impostato a 5 tramite la funzione `setReaders.` Nel caso in cui ci siano 5 thread "lettori" che possiedono già il lock in lettura non sarà possibile per un ipotetico 6° thread acquisire il read_lock. Tale thread dovrà quindi rimanere in attesa finchè almeno 1 degli altri thread non rilasci il lock in lettura.

2. Si supponga di effettuare la chiamata `L.setReaders(5)` su un certo lock L, mentre sono già presenti 7 lock in lettura; in tal caso i 7 lock in lettura vengono mantenuti, ma non sarà possibile acquisire ulteriori lock in lettura, finchè i lock attivi non siano almeno 4.

`void enableWriters(boolean enable).` Tramite questo metodo è possibile abilitare e disabilitare l'acquisizione di lock in scrittura sul `ReadWriteLockEvoluto` (se enable = true ⟶ la scrittura sulla risorsa sarà abilitata, se enable = false ⟶ la scrittura sarà disabilitata).

N.B. Quando si disabilita la possibilità di acquisire il lock in scrittura, eventuali write lock precedentemente acquisiti vengono mantenuti. Tutti i successivi thread che dovessero tentare di acquisire il lock in scrittura entreranno in uno stato di attesa, finchè la possibilità di acquisire il lock in scrittura non verrà riabilitata.

NON SPEGNERE IL PC A FINE ESAME

NON SPEGNERE IL PC A FINE ESAME

E' possibile fare uso del codice della classe ReadWriteLock esaminata durante il corso.

CI SONO DEI PUNTI AMBIGUI NELLA TRACCIA? COMPLETA TU

È parte integrante di questo esercizio completare le specifiche date nei punti non esplicitamente definiti, introducendo o estendendo tutte le strutture dati laddove si ritenga necessario, e risolvendo eventuali ambiguità.

POSSO CAMBIARE IL PROTOTIPO DEI METODI RICHIESTI? NO

Non è consentito modificare il prototipo dei metodi se questo è stato fornito. Potete aggiungere qualsivoglia campo e metodo di servizio, e qualsivoglia classe ausiliaria, ma NON variare l'interfaccia dei metodi pubblici già specificati.

CHE LINGUAGGIO DEVO USARE? JAVA 7 O SUCCESSIVO

Il linguaggio da utilizzare per l'implementazione è Java. *È* consentito usare qualsiasi funzione di libreria di Java 7 o successivi.

MA IL MAIN() LO DEVO SCRIVERE? E I THREAD DI PROVA? SOLO PER FARE IL TUO DEBUG

Non è esplicitamente richiesto di scrivere un `main()` o di implementare esplicitamente del codice di prova, anche se lo si suggerisce per testare il proprio codice prima della consegna.

NON SPEGNERE IL PC A FINE ESAME

ESERCIZIO 2 (Linguaggi di scripting. Punti 0-10)

Il comando bash `top -n 1 -b | tail -n +7` mostra alcune informazioni riguardanti l'utilizzo della CPU e della memoria da parte di utenti e processi.

Esempio 1

Un possibile output del comando `top -n 1 -b | tail -n +7` è:

```
  PID USER      PR  NI VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 2226 neutron   20   0  277852 124484  11932 S   6.2  0.1   6:30.42 neutron-linuxbr
 2573 nova      20   0  298392 136396   7400 S   6.2  0.1  10:43.46 nova-conductor
 2611 nova      20   0  445904 191720  13152 S   6.2  0.1   1:09.51 nova-api
21269 pacenza   20   0   41932   3920   3144 R   6.2  0.0   0:00.02 top
    1 root      20   0   37976   6140   4092 S   0.0  0.0   0:11.74 systemd
...   ...       ...  ...   ...    ...    ...  ...   ...      ...    ...
```

In questo esercizio bisogna realizzare uno script Perl, che verrà salvato nel file `top_stat.pl`. Lo script accetta un parametro opzionale su linea di comando che specifica il tipo di verifica che si vuole fare. L'argomento in input può essere `-c` (statistica sull'utilizzo della CPU in percentuale) o `-m` (statistica sull'utilizzo della memoria in percentuale).
Se non dovesse essere inserito alcun parametro, lo script, di default, restituirà informazioni riguardanti l'utilizzo del processore.

Ad ogni esecuzione, lo script dovrà mostrare su Standard Output, in ordine **decrescente,** l'utilizzo totale del processore o della memoria in percentuale, raggruppato per ogni singolo utente.
Bisogna visualizzare il tutto seguendo il formato specificato nella sezione **Formato Output – Standard Output.**

Successivamente sarà necessario salvare l'output su un **file** di log che chiameremo `top_stat.log`.
Bisogna visualizzare il tutto seguendo il formato specificato nella sezione **Formato Output – Log File.**

BONUS (max 1pt su parte Perl): A parità di utilizzo delle risorse richieste, si richiede di ordinare in primo luogo per valore decrescente di utilizzo delle risorse, ed in secondo luogo anche per ordine lessicografico degli utenti.

NON SPEGNERE IL PC A FINE ESAME

Esempio Output - Standard Output

pacenza@pcino:~$ perl script.pl -c

```
USER --- %CPU
root --- 23.6%
pacenza --- 11.8%
clamav --- 0%
cvsd --- 0%
```

pacenza@pcino:~$ perl script.pl -m

```
USER --- %MEM
root --- 4.5%
pacenza --- 0.8%
clamav --- 0%
cvsd --- 0%
```

N.B.: Le stampe sono ordinate prima per valore decrescente di utilizzo delle risorse e a parità di uso delle risorse in ordine lessicografico degli utenti (si notino gli utenti `clamav` e `cvsd` che usano entrambi lo `0%` delle risorse, di conseguenza sono ordinati lessicograficamente).
È **obbligatorio** ordinare in ordine **decrescente** l'uso dei processi, mentre è facoltativo e rientra nel **bonus** ordinare anche **lessicograficamente**.

Esempio Output - Log File

```
USER --- %CPU
root --- 23.6%
pacenza --- 11.8%
clamav --- 0%
cvsd --- 0%
####################
USER --- %MEM
root --- 4.5%
pacenza --- 0.8%
clamav --- 0%
cvsd --- 0%
####################
```

Formato Output - Standard Output (DA RISPETTARE RIGOROSAMENTE)

Per ogni utente `u` mostrato nell'output del comando `top` (si veda colonna `USER`), sia `s` la quantità totale di risorsa consumata da `u`.

Stampare sulla prima riga:

`USER - - - %CPU`

Oppure

`USER - - - %MEM`

Successivamente stampare ogni utente su una riga con la quantità di risorsa utilizzata separata da 3 trattini `- - -`

u_1 - - - s_1%
u_2 - - - s_2%
u_3 - - - s_3%
... ...

Formato Output - Log File (DA RISPETTARE RIGOROSAMENTE)

Si deve stampare su file, in modalità `append`, l'output di ogni esecuzione dello script.
Ogni nuova stampa deve essere separata dalla precedente da 20 `#`

```
USER - - - %MEM
u1 - - - s1%
u2 - - - s2%
u3 - - - s3%
... ...
####################
USER - - - %CPU
u1 - - - s1%
u2 - - - s2%
u3 - - - s3%
... ...
####################
USER - - - %CPU
u1 - - - s1%
u2 - - - s2%
u3 - - - s3%
... ...
####################
```