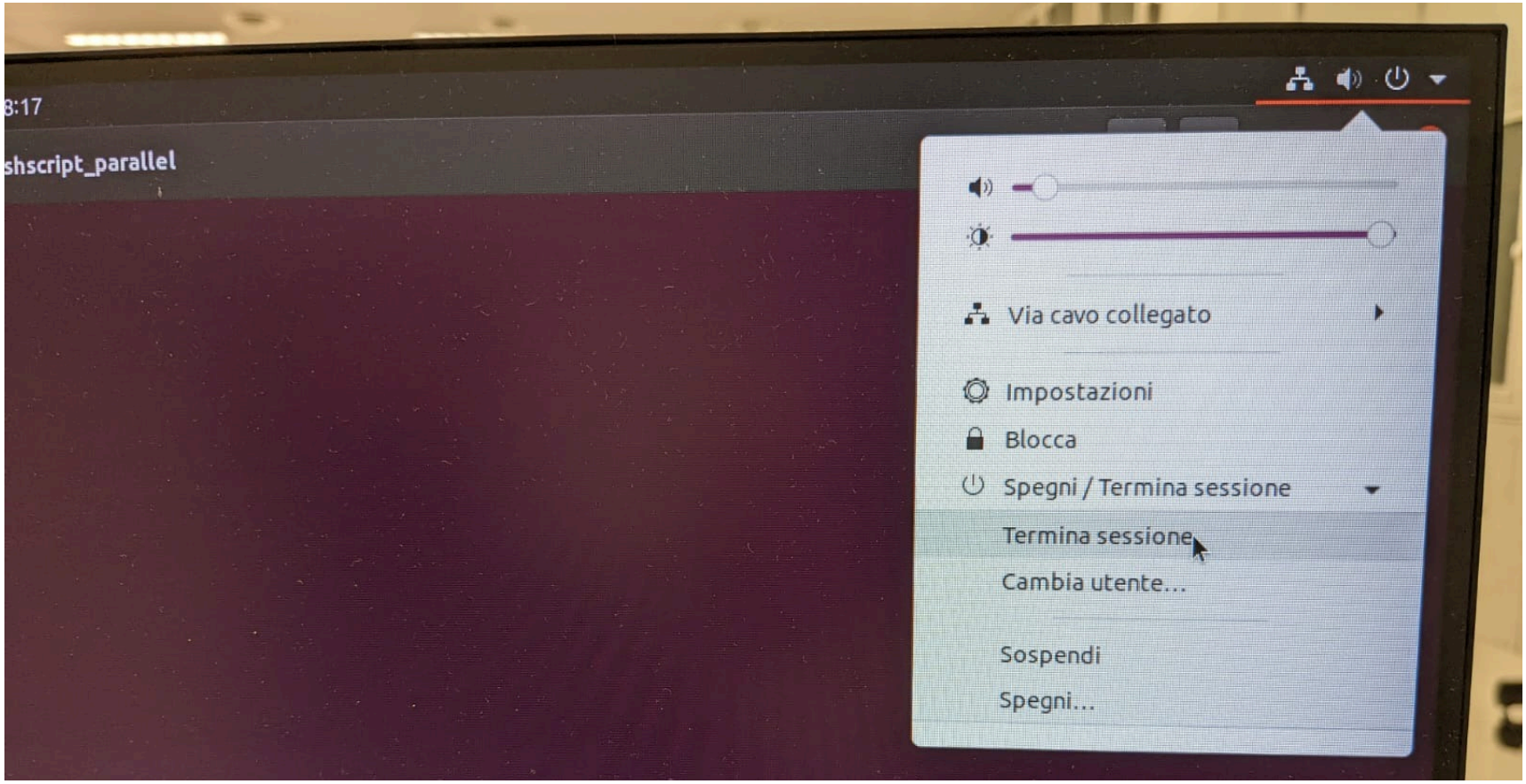
Corso di Sistemi Operativi e Reti

Corso di Sistemi Operativi

Prova scritta - Giugno 2024

ISTRUZIONI:

1. **Rinomina** subito la cartella chiamata "Cognome-Nome-Matricola" che hai trovato sul Desktop e in cui hai trovato questa traccia, sostituendo "Cognome" "Nome" e "Matricola" con i tuoi dati personali e **lasciando i trattini senza spazi**; se hai un doppio nome oppure un doppio cognome dovrai chiamare la cartella come in questo esempio:
    a. DeLuca-MarcoGiovanni-199999
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 fai "**Termina Sessione/Logout**", ma lascia la postazione mantenendo il PC acceso.
5. **E' tua diretta responsabilità** garantire l'integrità del tuo elaborato, anche in caso di assenza di corrente. **Salva spesso il tuo lavoro**



e NON spegnere il PC.

SALVA SPESSO

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 nuove strutture dati, o estendendo quelle preesistenti laddove si ritenga necessario, risolvendo eventuali ambiguità. Si può cambiare il codice dei metodi esistenti dove serve.

POSSO CAMBIARE IL PROTOTIPO DEI METODI RICHIESTI O DI QUELLI ESISTENTI? NO

Non è consentito modificare il prototipo dei metodi pubblici di una classe se questi sono stati forniti. Potete aggiungere qualsivoglia campo e metodo privato, e qualsivoglia classe ausiliaria, ma NON variare l'interfaccia dei metodi pubblici già specificati. Analogamente, i metodi esistenti possono essere modificati nel loro codice, ma non se ne deve cambiare l'interfaccia, il risultato finale o il significato.

CHE LINGUAGGIO POSSO USARE? PYTHON 3.X

Il linguaggio da utilizzare per l'implementazione è Python 3.6 o successivo. Ricorda che l'operatore di formattazione `f` (esempio, `f"Ciao sono la stringa {testo}"`) è disponibile solo dalla versione 3.6 di Python in poi, ma può essere sostituito con `"Ciao sono la stringa %s" % testo`

POSSO CONSENTIRE SITUAZIONI DI RACE CONDITION NEL MIO CODICE? NO
POSSO CONSENTIRE SITUAZIONI DI DEADLOCK NEL MIO CODICE? NO
POSSO CONSENTIRE ALTRE SITUAZIONI DI BLOCCO TOTALE NEL MIO CODICE, TIPO NESTED LOCKOUT, LIVELOCK O ALTRO? NO
POSSO CONSENTIRE SITUAZIONI DI STARVATION NEL MIO CODICE? SI, tranne quando ti viene chiesto esplicitamente di rimuoverle

MA IL MAIN() LO DEVO AGGIORNARE? E I THREAD DI PROVA? SI

E' obbligatorio implementare esplicitamente il codice di prova oppure modificare il codice di prova pre-esistente, e accertarsi che giri senza errori prima della consegna.

ESERCIZIO 1 - PROGRAMMAZIONE MULTITHREADED

(Punteggio minimo richiesto 18/30. Pesa per ⅔ del voto finale)

Punto 1

Modifica il codice in maniera tale che si possano creare delle istanze di `AmbienteGioco` in cui la dimensione predefinita del tavolo di gioco si possa scegliere attraverso un parametro.
Es. deve essere possibile creare ambienti di gioco come `a = AmbienteGioco(5)`
in cui i tavoli formati sono fatti di 5 giocatori.

Punto 2

Rendi parametrico il numero di thread giocatori che vengono creati (linea 132) e il tempo di gioco di ciascuno di essi (time.sleep(5) sul rigo 114). Dopodiché riduci il numero di giocatori e aumenta il loro tempo di gioco. Noterai che i tavoli di gioco si formano molto più lentamente.

Fai in modo che se il ThreadAbbinatore non trova gli N partecipanti necessari entro S secondi, i partecipanti mancanti vengono assegnati imponendo un ID fittizio negativo. Per ottenere ciò puoi usare la versione di `get` che prevede un timeout di attesa. Ad esempio `queue.get(timeout=S)` provoca l'eccezione chiamata `queue.Empty` qualora in coda non sia disponibile un elemento entro `S` secondi.

Esempio:

Immagina che S=5 secondi. Se in coda_richieste_gioco ci sono solo i due giocatori di 13 e 17 in attesa, trascorsi 5 secondi senza che arrivino altre richieste di partecipare a un tavolo, verrà composto il tavolo [13,17,-1,-2]

Punto 3

Supponi che non ci sia una dimensione fissa del tavolo di gioco, ma che ciascun giocatore possa dire che vuole partecipare a tavoli da 2, da 3, da 4 o da 5 persone. Modifica il codice in maniera tale che si possano organizzare partite dove ciascun giocatore viene assegnato a un tavolo della dimensione desiderata.

Ad esempio, se il giocatore ID1 chiede di partecipare a una tavolo da 3, esso aspetterà finchè il Thread Abbinatore non riuscirà a creare una partita formata da ID1 e da altri due giocatori che intendo partecipare a un tavolo da tre; i giocatori che intendono partecipare a tavoli da 5 persone attenderanno che si formino tavoli da 5 persone, ecc. ecc.

Esempio:
In coda_richieste_gioco ci sono i giocatori ID=2 e ID=5 che vogliono formare un tavolo da 2, il giocatore ID=7 che vuole partecipare a un tavolo da 3 e i giocatore ID=8 e ID=9 che vogliono partecipare a un tavolo da 4.

Si dovrà formare un tavolo da due così composto: [2,5]

Mentre il giocatore 7 dovrà aspettare che si formi un tavolo da 3 con altri due thread; invece i giocatori 8 e 9 attenderanno ulteriori due giocatori necessari a completare un tavolo da 4.

Punto 4
Prova a reimplementare il codice eliminando la barriera, e cambiando la forma di comunicazione tra Giocatori e Abbinatori.

Punto 5

Reingegnerizza il codice in maniera tale che: 1. i ThreadAbbinatori vengono avviati nel costruttore di AmbienteGioco; 2. si possa dire quanti ThreadAbbinatori si vogliono creare.

SALVA SPESSO

ESERCIZIO 2 - LINGUAGGI DI SCRIPTING

(Punteggio minimo richiesto 18/30. Pesa per ⅓ del voto finale)

Scrivi uno script Perl dal nome `copiato.pl` che riceve come parametri sulla linea di comando *i soli nomi* di due file da cercare sul Desktop dell'utente corrente. Lo script deve restituire il numero totale di linee e caratteri dei due file dopo aver rimosso eventuali linee duplicate da ciascun file. Infine deve stampare il numero di linee in comune ai due file già ripuliti dalle ripetizioni.

Esempio:
supponiamo che lo script venga invocato come segue

```
./copiato.pl file1.txt file2.txt
```

e che il contenuto dei file sia il seguente

file1.txt

```
ciao
questo è il contenuto di file1
fine.
ciao
```

file2.txt

```
ehi
    questo è il contenuto di file2
ciao
fine.
```

[segue nella pag. successiva]

lo script dovrebbe produrre in output

```
file1.txt
linee: 3
caratteri: 43
file1.txt
linee: 4
caratteri: 46
linee in comune: 2
```

[segue nella pag. successiva]

Si noti che le linee di file1.txt sono 3 in quanto il secondo "ciao" è stato rimosso poiché si tratta di una linea duplicata nel file stesso. Allo stesso modo, le linee in comune tra file1.txt e file2.txt sono 2 poiché, dopo aver rimosso i duplicati dei rispettivi file, rimangono in comune le linee "ciao" e "fine.".