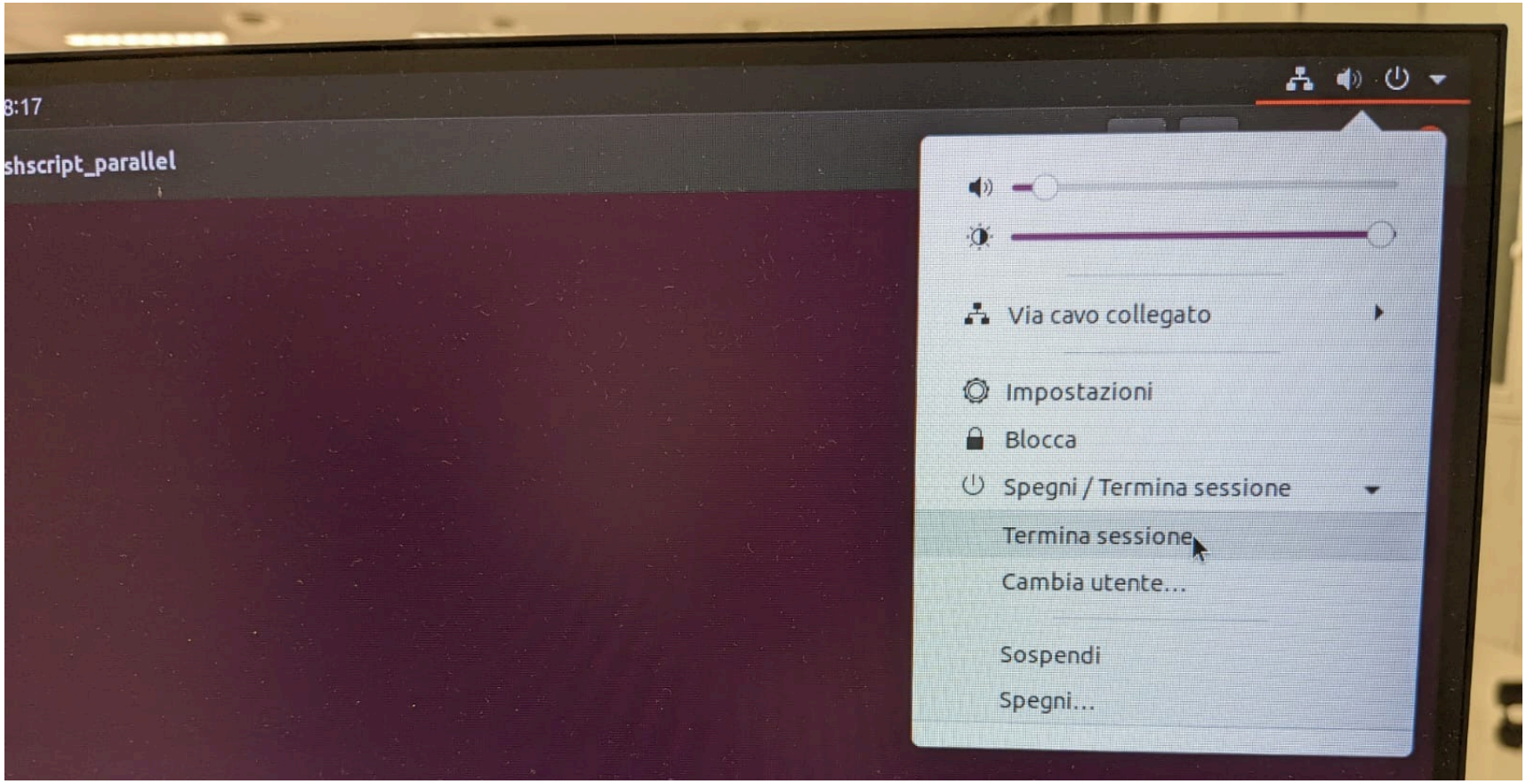
Corso di Sistemi Operativi

Corso estinto di Sistemi Operativi e Reti - Modulo Sistemi Operativi

Prova scritta - Gennaio 2025

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? DECIDI TU COSA FARE

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

Il codice che scriverai è pensato per essere usato come un modulo di libreria. Un modulo di libreria potrebbe essere usato da altri programmatori, i quali si aspettano di trovare una specifica interfaccia.

Non è quindi consentito modificare il prototipo dei metodi pubblici di una classe se questi sono stati forniti. Puoi 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

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)

Svolgi tutti i punti della traccia in un unico file che chiamerai `SOLUZIONE.py`

-> Qualsiasi altro nome di file verrà IGNORATO <-

Punto 1.

Osserva che ogni mossa di un Automobile consiste nello scegliere se muoversi in una casella libera in una certa direzione. Aggiungi, tra le mosse consentite, la possibilità che un'automobile possa decidere di lasciare un simbolo `'.'` in una casella contenente uno spazio `' '` nella casella immediatamente dietro all'automobile stessa. Il simbolo `'.'` è volto ad ostacolare eventuali auto che seguono l'auto corrente. Poiché le auto sono programmate per spostarsi solo negli spazi liberi, il punto dove è posizionato `'.'` diventerà un ostacolo. Quando immediatamente dietro all'auto non c'è uno spazio, non deve essere possibile fare questo nuovo tipo di mossa.
Per "dietro all'auto" si intende la colonna immediatamente a sinistra dell'auto stessa.

Esempio:

Prima:

```
############################
           K         A    C
```

Dopo:

```
############################
           K        .A    C
```

Punto 2.
Modifica il visualizzatore in maniera tale da stampare lo stato della corsa ogni volta che c'è un cambio di turno tra una partecipante e un altro, anziché a intervalli regolari. Spetta a te determinare come rilevare un passaggio di turno da un partecipante a un altro.

Punto 3.
Aggiungi un thread Penalizzatore. Il thread Penalizzatore sceglie ogni 5 secondi una vettura a caso e la blocca per 3 secondi. Nel frattempo tutte le altre auto devono poter continuare a muoversi. Decidi tu cosa fare se il penalizzatore sceglie un'auto che è di turno proprio in quel momento.

Punto 4.
Modifica il meccanismo di passaggio di turno in maniera tale da far ruotare il turno delle automobili come in un gioco a dadi tradizionale. Ad esempio, se ci sono solo tre auto A, B e C, il passaggio del turno di gioco deve seguire la sequenza A -> B -> C -> A -> B -> C ….

Punto 5.
Fai in modo che il gioco non termini appena il vincitore taglia il traguardo. Dai la possibilità a tutti i giocatori di terminare e alla fine della gara stampa l'esatto ordine di arrivo.

SALVA SPESSO

ESERCIZIO 2 - LINGUAGGI DI SCRIPTING

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

Scrivere uno script perl chiamato `elabora_file.pl` che si comporti come segue:

1. Stampi su STDOUT tutti i file `.py` **presenti in una qualsiasi sottocartella[1] della home dell'utente corrente;**
2. Dia diritti di esecuzione ai file `.py` **presenti nella sola cartella corrente** per l'utente corrente;
3. Stampi su STDOUT i percorsi dei file, **presenti in una qualsiasi sottocartella[1] della home dell'utente corrente**, per cui esiste almeno un altro file con lo stesso nome. Stampare i percorsi di tutti gli omonimi.

   Esempio per il punto 3:

   L'utente `esame` esegue lo script. Nella home dell'utente esistono i file `/home/esame/file1.pdf`, `/home/esame/file2.py`, `/home/esame/dir1/file1.pdf`

   lo script dunque stampa

   ```
   /home/esame/file1.pdf
   /home/esame/dir1/dir2/file1.pdf
   ```

[1] A qualsiasi profondità