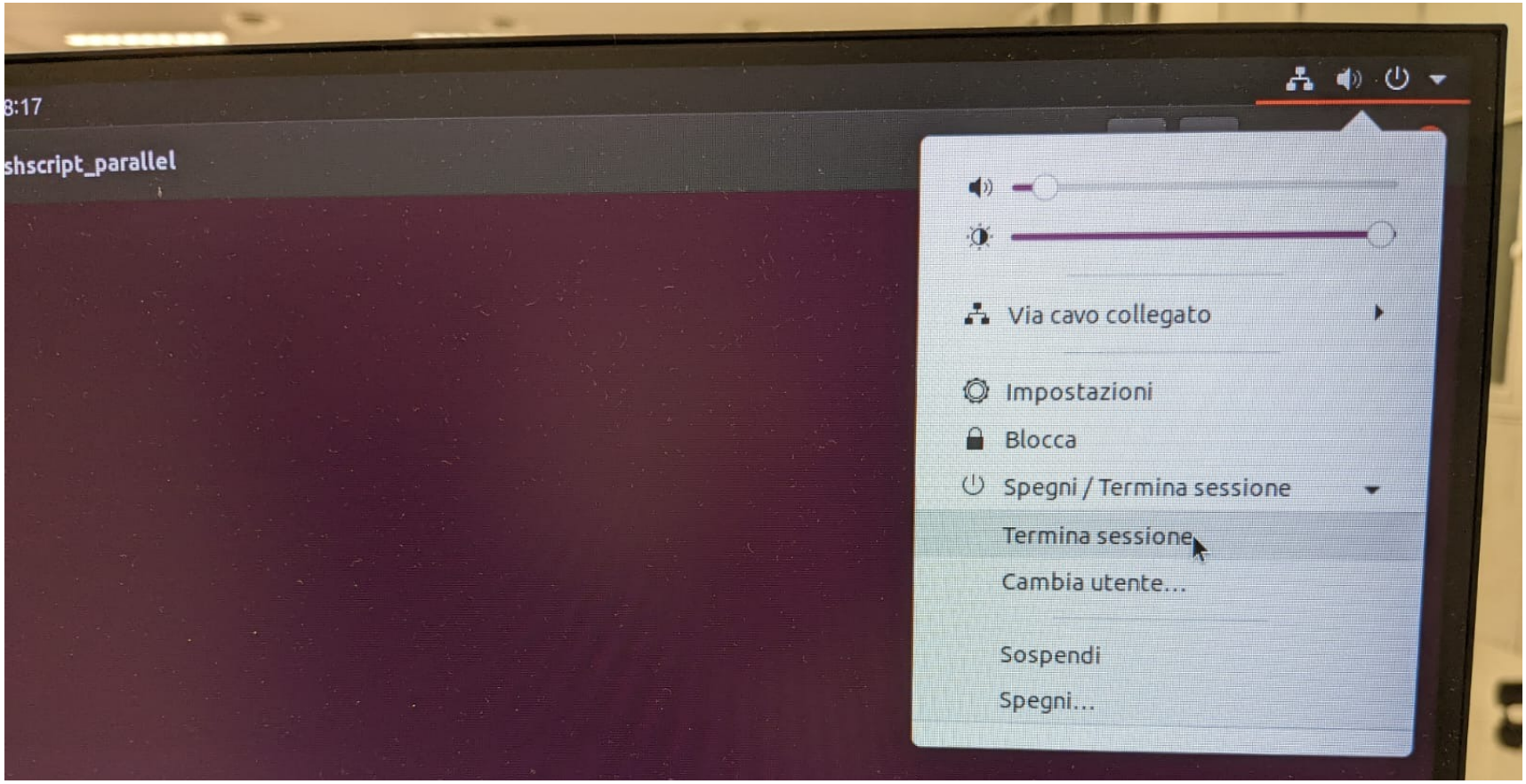
Corso di Sistemi Operativi e Reti

Corso di Sistemi Operativi

Prova scritta - 26 Gennaio 2023

ISTRUZIONI PER CHI È IN PRESENZA:

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**”, lascia la postazione



e NON spegnere il PC.

SALVA SPESSO

ISTRUZIONI PER CHI SI TROVA ONLINE:

1. **Questo file contiene il testo che ti è stato dato ieri, incluso il codice;**
2. **Mantieni a tutto schermo** questo file per tutta la durata della prova; puoi scorrere liberamente tra le sue pagine, ma non puoi cambiare applicazione;
3. **Firma** preliminarmente il foglio che userai per la consegna con nome cognome e matricola;
4. **Svolgi** il compito; puoi usare solo carta, penna e il tuo cervello;
5. **Aiutati** con i numeri di linea per indicare le eventuali modifiche che vorresti fare al codice che ti è stato dato.
6. **Alla scadenza** termina *immediatamente* di scrivere, e attendi di essere chiamato, pena l'esclusione dalla prova;
7. **Quando è il tuo turno** mostra il foglio ben visibile in webcam, e poi metti una foto dello stesso foglio in una chat privata Microsoft Teams con il prof.

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 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. Analogamente, i metodi esistenti possono essere modificati nel loro codice, ma non se ne deve cambiare 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 del codice di prova oppure modificare il codice di prova pre-esistente, e accertarsi che giri senza errori prima della consegna.

ESERCIZIO 1 - PROGRAMMAZIONE MULTITHREADED

Punto 1

Come primo requisito, la classe `ExtendedBarrier` dovrà estendere il codice iniziale di `Barrier` con i metodi `finito` e `aspettaEbasta(self)`. Il primo metodo incrementa di uno i threadArrivati ed esce; il secondo metodo si mette in attesa che i threadArrivati raggiungano la soglia prescritta per poi uscire quando questa condizione si verifica, ma senza incrementare il numero di threadArrivati.

Punto 2

Sempre agendo sulla classe `ExtendedBarrier`, fattorizza il metodo `wait`, sfruttando i metodi `finito` e `aspettaEbasta` che hai appena implementato.

Punto 3

Progetta e implementa la classe `DoppiaBarriera`. La classe deve consentire di incrementare e attendere due soglie in contemporanea: la soglia S0 e la soglia S1. Il costruttore di `DoppiaBarriera` riceve due valori di soglia `n0` e `n1` che saranno rispettivamente associati alla soglia S0 e alla soglia S1.

In altre parole questa classe deve fornire i metodi:

`finito(self,numSoglia)`

Incrementa i threadArrivati sulla soglia `numSoglia`, dove `numSoglia` può valere 0 oppure 1 a seconda che si voglia scegliere la soglia S0 o la soglia S1.

`aspettaEbasta(self,numSoglia)`

Attende che i thread arrivati su `numSoglia` raggiungano la soglia impostata, dove `numSoglia` può valere 0 oppure 1, per poi uscire.

`wait(self,numSoglia)`

Incrementa i threadArrivati associati alla soglia `numSoglia`, quindi attende che i thread arrivati su `numSoglia` raggiungano la soglia prescritta per poi uscire.

`waitAll(self)`

Incrementa di uno i thread arrivati su entrambe le soglie e aspetta che *entrambe* le soglie siano raggiunte dal numero di thread prescritti.

SALVA SPESSO

ESERCIZIO 2 - LINGUAGGI DI SCRIPTING

Scrivi uno script Perl dal nome **`ricettario.pl`** che consenta di esplorare un catalogo di ricette. Ogni ricetta è contenuta in un file all'interno della cartella `RICETTE` che è stata fornita. Ciascuno dei file che troverai nella cartella è denominato come `nome_ricetta.txt` e contiene: *a)* un elenco di **ingredienti** e *b)* lo svolgimento della ricetta stessa. La parte *a)* del file è contrassegnata da un rigo con la parola **Ingredienti**, mentre l'inizio della parte *b)* è contrassegnato da un rigo con la parola **Preparazione**.

Esempio:

Il contenuto del file denominato `gyoza.txt` potrebbe essere il seguente:

Ingredienti

250 g di farina 00

150 g d'acqua a temperatura ambiente

1 pizzico di sale

Per il ripieno:

150 g di polpa di maiale macinata

50 g di gamberetti sgusciati

(...)

Preparazione

Disponete a fontana la farina su di un piano di lavoro e dopo aver messo nel mezzo l'acqua ed il sale, cominciate ad impastare energicamente.

Lavorate il composto fino a che si sarà formata una palla liscia ed omogenea simile all'impasto della pizza.(...)

Lo script deve poter essere invocato nel seguente modo

```
ricettario.pl INGREDIENTE [ INGREDIENTE]...
```

passando come parametro **uno o più** ingredienti separati da spazio.
Lo script dovrà individuare le ricette che contengono **almeno uno** dei suddetti ingredienti. Appena viene trovata una ricetta che contiene un ingrediente ricevuto come parametro, si dovrà stampare in STDOUT il nome della ricetta e il rigo contenente l'ingrediente individuato. Ogni ricetta in elenco dovrà essere associata ad un numero progressivo.
A quel punto, lo script dovrà rimanere in attesa finché non riceverà in STDIN il numero corrispondente ad una ricetta in elenco: stamperà quindi in STDOUT lo svolgimento della ricetta stessa. Se si riceve in input la parola `END` lo script deve terminare immediatamente. Decidi tu come gestire eventuali casi di errore (numeri fuori intervallo, ecc. ecc.).
***RICORDA**: puoi utilizzare delle variabili all'interno delle espressioni regolari scrivendo, ad esempio,* `/$variabile/`.

Esempio:

Lo script potrebbe essere lanciato nel seguente modo:

```
./ricettario.pl gamberetti maiale
```

e in questo caso potrebbe produrre in output:

```
1- GYOZA: 150 g di polpa di maiale macinata
2- LASAGNA: 250 g di carne di maiale tritata
(...)

>> 1(INPUT UTENTE VIA <STDIN>)
```

Disponete a fontana la farina su di un piano di lavoro e dopo aver messo nel mezzo l'acqua ed il sale, cominciate ad impastare energicamente.
Lavorate il composto fino a che si sarà formata una palla liscia ed omogenea simile all'impasto della pizza.(...)