

# Corso di Sistemi Operativi e Reti

## Corso di Sistemi Operativi

Prova scritta 20 Giugno 2022

### ISTRUZIONI PER CHI È IN PRESENZA:

1. **Rinomina** 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**; 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 lascia la postazione facendo logout,

**senza 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

Si deve modificare la classe `PivotBlockingQueue` in maniera tale da poter specificare quanti pivot ci sono in ciascun momento. Per realizzare questo scopo, introduci il metodo `setPivotNumber(self, n : int)`, che imposta il numero di pivot al valore `n` (`n` deve essere almeno 1 e al massimo `N-1`, dove `N` è la dimensione della coda).

Il numero di pivot impostato in un certo momento influenza il numero di rimozioni di elementi prescritte. Dovrai fare in modo che il cambio della quantità di pivot influenzi i metodi `take()` e `put()` per come spiegato di seguito. Facciamo un esempio e supponiamo che il numero di pivot venga impostato a due, anziché uno come nel codice preesistente.

Se i pivot presenti vengono impostati a due, e non più a uno come nel codice esistente, dovrai fare in modo che `take()` elimini due pivot prima di restituire l'elemento richiesto, e cioè *il primo massimo/minimo e il secondo massimo/minimo*. `take()` dovrà invece bloccarsi fintantochè in coda non ci siano almeno tre elementi (i due pivot da rimuovere più l'elemento da estrarre). L'operazione di `put` resta invece invariata e rimuove un solo pivot nel caso in cui la coda sia piena.

E' a tuo carico stabilire come gestire opportunamente il caso in cui ci sono due o più pivot di valore uguale e cioè due o più massimi/minimi di pari valore.

### Punto 2

Si estenda la classe `PivotBlockingQueue` con il metodo `doubleTake()`. Tale metodo preleva e restituisce una coppia di elementi anziché uno solo, e si blocca se in coda non sono presenti almeno due elementi + i pivot. Decidi tu quale sia la codifica migliore per restituire una coppia di interi anziché un solo valore intero. Nota che per risolvere questo punto è necessario aver implementato il punto 1.

### Punto 3

Introduci un metodo `waitFor(self, n)` che va in attesa bloccante finché la somma degli elementi presenti nella coda non supera il valore `n` specificato, uscendo se la condizione è invece verificata.

# SALVA SPESSO

## ESERCIZIO 2, TURNO 1 - PERL

Si scriva uno script Perl dal nome `documentazione.pl` che estragga alcune informazioni utili dalla documentazione di un certo comando shell ricevuto come parametro.

Lo script deve essere invocato nel seguente modo

```
./documentazione.pl OPZIONE comando
```

I parametri `OPZIONE` e `comando` sono entrambi obbligatori. `comando` rappresenta il nome di un comando disponibile nella proprio shell.

Assumeremo che `comando` abbia delle opzioni **brevi**, le quali iniziano con un solo trattino "-", mentre considereremo come **lunghe** tutte le opzioni che iniziano con un doppio trattino "--").

I possibili valori del parametro `OPZIONE` sono i seguenti:

- **-n**

quando viene specificata questa opzione bisogna produrre in `STDOUT` il numero di opzioni **brevi** del comando specificato dal parametro `comando`

**Esempio:**

Lo script potrà essere lanciato nel seguente modo:

```
./documentazione.pl -n rm
```

e dovrà produrre in `STDOUT` la stringa:

6 opzioni brevi

- **-o**

Se viene specificata questa opzione, bisogna stampare su un FILE dal nome **opzioni\_brevi.log** l'elenco delle opzioni **brevi** con relativa descrizione. La stampa dovrà essere fatta ordinando alfabeticamente la descrizione delle opzioni.

**Esempio:**

Lo script potrà essere lanciato nel seguente modo:

```
./documentazione.pl -o rm
```

e dovrà produrre in output:

```
-v: explain what is being done  
-f: ignore nonexistent files and arguments, never prompt  
-i: prompt before every removal  
-I: prompt once before removing more than three files, or when removing recursively; less  
intrusive than -i, while still giving protection against most mistakes  
-r: remove directories and their contents recursively  
-R: remove directories and their contents recursively  
-d: remove empty directories
```