Sistemi di Calcolo 2 (A.A. 2019-2020 e seguenti)

Prova congiunta Sistemi di calcolo 12 CFU - Seconda parte (A.A. 2017-2018 e precedenti)

Appello straordinario - 26 Gennaio 2021

Tempo a disposizione: 1h 45m.

Attenzione: assicurarsi di compilare il file **studente.txt** e che il codice prodotto non contenga **errori di compilazione**, pena una non correzione dell'elaborato.

Regole Esame

- Domande ammesse
  Le domande possono riguardare solo la specifica dell'esame e la struttura di alto livello del codice, nessuna domanda può riguardare singole istruzioni.
- Oggetti vietati
  I seguenti oggetti non devono essere presenti sulla scrivania, né tantomeno usati: smartphone, smartwatch, telefonini, tablet, portatili, dispositivi di archiviazione USB, copie cartacee della dispensa, astucci e qualsiasi forma di libri ed appunti. **Chi verrà sorpreso ad usare uno di questi oggetti verrà automaticamente espulso dall'esame.**
- Modalità di risposta
  Le risposte alle domande di teoria vanno fornite nei file txt indicati. Il professore fornisce fogli per appunti e dove fornire illustrazioni o codice utile ad integrazione delle risposte di teoria. Qualsiasi altro foglio portato dallo studente non può essere usato.
- Azioni vietate
  È assolutamente vietato comunicare in qualsiasi modo con gli altri studenti. **Chi verrà sorpreso a comunicare con gli altri studenti per la prima volta verrà richiamato, la seconda volta verrà invece automaticamente espulso dall'esame**.

Teoria 1 - Deadlock (rispondere nel file teoria1.txt)

Descrivere le condizioni necessarie e sufficienti per il verificarsi del deadlock e gli approcci per la gestione del deadlock (non mi riferisco a semafori, monitor….)

Teoria 2 - Algoritmi di concorrenza (rispondere nel file teoria2.txt)

Considerate il seguente algoritmo del panettiere "modificato":

```
Initially
/* global info */
boolean choosing[N]   = {false, ..., false};
integer num[N] = {0, ..., 0};
/* local info */
int i = <entity ID>;  // i ∈ {0, 1, ..., N-1}
repeat
1     NCS
2     choosing[i] := true
3     num[i] := 1+ max {num[j] : 0 <= j < N}
4     choosing[i] := false
5     for j := 0 to N-1 do begin
6           while choosing[j] do skip
7           while num[j] != 0 OR {num[j], j } < {num[i],i} do skip
8     end
9     CS
10    num[i] := 0
```

forever

L’algoritmo del panettiere è stato modificato alla riga 7 sostituendo l’operatore logico AND con l’operatore logico OR

A. Descrivere cosa comporta tale modifica nell’esecuzione del programma
B. Quale o quali sono le problematiche che si creano legate alle proprietà di Mutua Esclusione, No-Deadlock e No-Starvation? Se e quali vengono violate? Motivare la risposta.

Programmazione - Realizzazione di un server multi-thread a concorrenza controllata

Un sistema offre un servizio di immagazzinamento e recupero delle informazioni, dando maggiore priorità all'informazione più recente. Un client può quindi richiedere al server di immagazzinare un dato (generato casualmente), o di recuperare l'ultimo elemento inserito (anche se l'elemento più recente potrebbe essere stato inserito da un altro client), a seconda del desiderio dell'utente, che ha a disposizione 3 comandi: w (inserisci), r (recupera), q (esci).

Il server multi-thread predispone una listening socket per accettare connessioni in ingresso, e ad ogni connessione da parte di un client il server crea un processo figlio che si occuperà di gestire la richiesta. Il server riceve un messaggio di lunghezza fissa (una struttura contenente il comando e un valore: il valore casuale da scrivere, oppure 0) e dopo aver gestito il comando (se 'q' termina immediatamente il thread) risponde con un intero (il valore richiesto se il comando era 'r', 1 per conferma se il comando era 'w'). I dati vengono memorizzati all'interno di un buffer gestito come una pila. Quando tale buffer è pieno i thread che vogliono scrivere vengono bloccati finché non si libera una posizione. Il livello di concorrenza nel server viene gestito tramite semafori unnamed su cui operano i thread che gestiscono le connessioni in ingresso accettate dal server.

Completare le parti identificate all'interno del file server.c

Obiettivi dell'esercizio

1. Gestione di connessioni in ingresso con paradigma multi-thread
2. Gestione del livello di concorrenza interno tramite semafori unnamed

Altro

- **i commenti nel codice contengono molte informazioni utili per lo svolgimento della prova, si consiglia quindi di tenerli in debita considerazione**
- il file `dispensa.pdf` contiene una copia della dispensa *Primitive C per UNIX System Programming* preparata dai tutor di questo corso
- il file `raccomandazioni.pdf` contiene una serie di considerazioni sugli errori riscontrati più di frequente

---