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

Secondo Appello - 20 Luglio 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 - Processi e Thread (rispondere nel file teoria1.txt)

Spiegare cosa è Inter-Process Communication e motivare la necessità di meccanismi di IPC.

Fornire illustrazioni cartacee a supporto della risposta, se ritenuto utile.

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

Considerate il seguente algoritmo di Dijkstra “modificato”:

```
Initially
     /* global info */
     boolean interested[N] = {false, ..., false};
     boolean passed[N] = {false, ..., false};
     int k = <any> ;       // k ∈ {0, 1, ..., N-1}
     /* local info */
     int i = <entity ID>;  // i ∈ {0, 1, ..., N-1}
repeat
   1 <Non Critical Section>;
   2 interested[i] = true;
   3 while (k != i) {
   4     passed[i] = false;
   5     if (!interested[k]) then k = i;
   6 }
   7 passed[i] = true;
   8 for j in 0 ... N-1 except i do
   9     if (!passed[j]) then goto 3;
  10 <critical section>;
  11 passed[i] = false; interested[i] = false;
forever
```

L’algoritmo di Dijkstra è stato modificato alla riga 9 negando la condizione

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 - ReverseServer multi processo

Un processo client invia, tramite socket, messaggi a un ReverseServer. Quest'ultimo è un processo (file `server.c`) che, per ogni connessione ricevuta, genera un processo figlio per la gestione delle comunicazioni. Per ogni messaggio ricevuto, il ReverseServer invia al client, tramite socket, la stringa ricevuta invertita. Le funzioni di invio e ricezione su socket andranno implementate nel file `io.c`. Lo studente deve completare e compilare i file `server.c, client.c` e `io.c`. Per testare indipendentemente i file `server.c` e `client.c`, possono essere utlizzati i file `profserver` e `profclient` inclusi nella distribuzione.
Obiettivi principali

1. Gestione multi-processo: creazione/terminazione processi figlio
2. Comunicazione via socket
3. Invio/Ricezione messaggi di lunghezza variabile (con condizione di terminazione)

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

---