

Tema d'esame
di Sistemi Operativi I e II e Laboratorio
18 novembre 2008
Docenti Luigi Lavazza, Matteo Vaccari e Igor Nai Fovino

Esercizio 1. (6 punti)

Si consideri un sistema in cui sono presenti tre processi uguali.

Ciascun processo esegue il seguente ciclo:

1. aspetta
2. se è abilitato prosegue, se no torna al punto 1
3. esegue una certa elaborazione (non importa quale)
4. stabilisce quali processi sono abilitati
5. torna al punto 1

L'abilitazione dell'*i*-esimo processo è indicata dalla *i*-esimo elemento del vettore *abil*, definito come segue:

```
#define N 3  
int abil[N];
```

Si realizzi in pseudo-codice C il sistema descritto, evitando corse critiche, deadlock e starvation. Si utilizzino i semafori per gestire la sincronizzazione. È possibile usare tutte le strutture dati e semafori che si ritiene opportuno.

NB: i tre processi sono uguali, quindi basta lo pseudo-codice di uno dei tre.

Dallo pseudo-codice deve emergere il principio della soluzione. Ulteriori dettagli sono inutili.

Esercizio 2. (3 punti)

Si definisca in pseudo-codice C un sistema in cui esistono due processi e in cui prima o poi si verifica sicuramente una condizione di deadlock.

Esercizio 3. (3 punti)

Si illustri (in non più di 10 righe) quali sono i meccanismi di MINIX che rendono possibili le chiamate di sistema. In altre parole, si illustri cosa succede nel sistema quando un processo utente esegue una chiamata di sistema.

Esercizio 4. (4 punti)

Si scriva un programma C che riceve in input il nome di due file, ed esegue la copia dei contenuti del primo sul secondo, sovrascrivendolo.

Esercizio 5. (4 punti)

Si descriva brevemente il funzionamento della chiamata di sistema `libreria malloc(3)`, spiegando soprattutto: il comportamento dal punto di vista del chiamante (argomenti, valori restituiti); il comportamento dal punto di vista dell'implementazione: quali strutture dati utilizza; dove vengono salvate queste strutture.

Esercizio 6 (4 punti)

Si realizzi uno script di shell che prenda come parametri:

- (1) un carattere alfanumerico
- (2) il nome di un file ASCII

Tale script dovrà eliminare dal file ASCII tutte le occorrenze del carattere specificato. Il tutto **SENZA USARE** il comando “tr”.

Suggerimento: potrebbe essere d’aiuto pensare al carattere da eliminare come ad un carattere speciale che “separi” porzioni di testo all’interno delle righe.

Esercizio 7 (3 punti)

Si descriva in dettaglio il meccanismo di controllo degli accessi ai file di Linux

Esercizio 8 (3 punti)

Si descriva la funzione del file fstab.