Corso di Laurea in Informatica

Esame di Sistemi Operativi
6 Luglio 2001

Istruzioni

- Scrivere Cognome, Nome, e Numero di Matricola sulla prima pagina di **ogni foglio**
- Scrivere le risposte alle domande relative al corso di Laboratorio di Sistemi Operativi su un **foglio separato**
- Tempo a disposizione: **3 ore**

Sistemi Operativi

1. Si mostri come implementare un semaforo binario con le regioni critiche.

2. In un sistema i processi possono accedere ad una risorsa condivisa in 3 modalità che chiamiamo A, B, e C. Per accedere alla risorsa in modalità X un processo invoca una funzione ACCESSO_X. Al termine dell'accesso alla risorsa in modalità X un processo invoca una funzione TERMINA_X. Le regole di accesso sono le seguenti: *nel momento in cui un ottiene il permesso di accedere alla risorsa in modalità A il numero di processi che stanno utilizzando la risorsa in modalità non deve eccedere la metà dei processi che stanno utilizzando la risorsa in modalità B o C.* Si utilizzino le regioni critiche per descrivere la politica di accesso alla risorsa condivisa.

3. Come si fa a stabilire se in un sistema con risorse ad istanze multiple ci troviamo in uno stato di deadlock? E se le risorse sono ad istanza singola?

4. Si fornisca un esempio di situazione in cui gli algoritmi SCAN e C-SCAN evadono una sequenza di richieste di accesso al disco rigido facendo attraversare alla testina lo stesso numero di cilindri.

Laboratorio di Sistemi Operativi

1. Cosa sono i pipe di tipo half-duplex (o unnamed) di UNIX e in cosa differiscono dai pipe FIFO (o named). Come funziona il collegamento tramite pipe di un comando che scrive su stdout con un comando che legge da stdin?

2. Quali sono i passi da seguire per la compilazione ed esecuzione di programmi Java che realizzino una comunicazione client-server basata su RMI?

3. Il problema dei 5 filosofi non ammette una soluzione semplice e robusta con i semafori ordinari. Viceversa, i semafori UNIX consentono una soluzione elegante. Si spieghi perchè e si abbozzi la soluzione.

4. Dato il metodo Java

```
public synchronized void Loop() {
 P1();
 try {
   wait(100);
 } catch(Exception e){}
 P2();
}
```

   lo si converta in uno equivalente che usi sleep() al posto di wait(). Spiegare perché questa seconda realizzazione è vulnerabile (nel senso delle race condition).