

**UNIVERSITA' DI BOLOGNA - CORSO DI LAUREA IN INFORMATICA
CORSO DI SISTEMI OPERATIVI - ANNO ACCADEMICO 2003/2004
COMPITO CONCORRENZA - 20 Settembre 2004**

Esercizio -1: essersi iscritti correttamente per svolgere questa prova.

Esercizio 0: Su entrambi i fogli, scrivere correttamente nome, cognome, matricola e posizione prima di svolgere ogni altro esercizio.

Esercizio 1

Si implementi, tramite monitor, uno stack a capacità limitata con chiamate:

- void push(Object o) - inserisce un elemento nello stack (attende se pieno)
- Object pop() - rimuove e restituisce un elemento dallo stack (attende se vuoto)
- Object top() - restituisce il prossimo elemento da rimuovere nello stack, senza rimuoverlo (attende se vuoto)
- boolean empty() - restituisce true se stack è vuoto, falso altrimenti
- void atleast(int n) - attende che nello stack ci siano almeno n elementi

Esercizio 2:

P1: S1.P(); S1.P(); S1.P(); S1.P(); S2.P(); S2.P(); print(A); S1.V();S1.V()

P2: S1.V();print(B);S1.V();S1.P();print(C);S1.V();S2.V();

P3: S1.V();print(D);S1.V();S1.P();print(E);S1.V();S2.V();

Scrivere tutte le possibili sequenze generate dall'esecuzione del programma composto dai processi P1,P2 e P3.

I semafori sono tutti inizializzati a zero. Sono possibili casi di deadlock? Se si, quali? Se no, perché?

Esercizio 3

Dato un sistema di supporto per message passing asincrono dotato delle chiamate asend e arecv scrivere un meccanismo a messaggi doppi a2send a2recv che operi nel modo spiegato qui di seguito. La a2send consente di spedire un messaggio in modo asincrono mentre la a2recv restituisce solo coppie di messaggi provenienti dallo stesso mittente. Quindi la a2recv deve sempre attendere due messaggi che rispettino le richieste del ricevente (mittente giusto, o ogni mittente se '*') provenienti dallo stesso mittente prima di potersi sbloccare.

Il problema è risolvibile senza fare uso di un processo server.