Nome_____ Cognome _____N. di matricola (10 cifre) _____ Riga ___ Col _____

UNIVERSITA' DEGLI STUDI DI BOLOGNA - CORSO DI LAUREA IN INFORMATICA
CORSO DI SISTEMI OPERATIVI - ANNO ACCADEMICO 2006/2007
CONCORRENZA - 17 Settembre 2007

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

Esercizio 0: Scrivere correttamente nome, cognome, matricola e posizione prima di svolgere ogni altro esercizio.

Esercizio 1: Scrivere un monitor con le seguenti P.E.:

```
void inserisci(T elem);
T estrai();
void ampiezza(unsigned int N);
```

La procedura "inserisci" aggiunge un elemento che verra' ricevuto da un processo che chiamera' la "estrai".

- piu' processi contemporaneamente possono chiamare la inserisci e la estrai.
- gli elementi vengono dati ai processi che chiamano la "estrai" nell'ordine in cui sono stati inseriti (con "inserisci"), i.e. la politica e' FIFO.
- La struttura ha capacita' limitata, inizialmente pari ad 1 elemento. Ogni processo che non puo' inserire o estrarre elementi deve attendere che l'operazione sia possibile.
- La funzione ampiezza cambia la capacita' della struttura. Qualora la capacita' venga ridotta e vi siano troppi elementi, i primi in attesa vengono eliminati (i.e. non verranno mai consegnati a processi che chiameranno la "estrai").

Esercizio 2: La primitiva *bloccante* **send2(msg m, pid rcv1, pid rcv2)** invia il messaggio **m** sia al processo **rcv1** che al processo **rcv2** e sospende il processo chiamante fino a quando **m** non sia stato ricevuto da entrambi. Per ricevere un messaggio inviato con la **send2**, un processo invoca la primitiva *bloccante* **msg receive2(pid snd)**. Infine, la **send2(m,rcv1,rcv2)** e' una primitiva *atomica*, in quanto il messaggio **m** viene consegnato atomicamente a entrambi i destinatari quando entrambi i destinatari hanno invocato la **receive2**. Nota: **non** e' ammessa la forma **receive2(*)**.

a) implementare la **send2** e la **receive2** usando le usuali primitive di comunicazione asincrona.

b) e' possibile implementare le usuali primitive di comunicazione asincrona usando la **send2** e la **receive2**, ma senza aggiungere un processo server? Motivare la risposta.

Esercizio 3: Nel piccolo regno di Theodorus vi e' un'unica strada circolare, un ponte a senso unico e due soli abitanti. Un abitante percorre sempre la strada in senso orario, l'altro la strada in senso antiorario. Per evitare di scontrarsi sul ponte, i due abitanti adottano il seguente protocollo.

```
int direzione = CLOCKWISE;
int changedir(int dir) { return (dir == CLOCKWISE ? COUNTERCLOCKWISE : CLOCKWISE); }
process vettura(int dir) {
    while(1) {
        sleep(random(10000)); /* compie il percorso da casa al ponte */
        while(dir != direzione);
        sleep(random(10));  /* attraversa il ponte impiegando un tempo casuale */
        direzione = changedir(direzione);
        sleep(random(10000)); /* compie il percorso dal ponte a casa */ }
```

a) il protocollo evita scontri sul ponte? (motivare bene)

b) il protocollo garantisce l'arrivo a casa in un tempo finito? (motivare bene)

c) il protocollo e' efficace nel minimizzare i tempi di attesa per l'accesso al ponte? In caso affermativo motivare la risposta. In caso negativo mostrare una possibile esecuzione in cui l'attesa sia eccessiva.

Nome_____ Cognome _____N. di matricola (10 cifre) _____ Riga ___ Col _____

UNIVERSITA' DEGLI STUDI DI BOLOGNA - CORSO DI LAUREA IN INFORMATICA
CORSO DI SISTEMI OPERATIVI - ANNO ACCADEMICO 2006/2007
PARTE GENERALE - 17 Settembre 2007

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

Esercizio 0: Scrivere correttamente nome, cognome, matricola e posizione prima di svolgere ogni altro esercizio.

Esercizio 1: Consideriamo un sistema caratterizzato da: 4 frame di memoria virtuale, condivisi fra tutti i processi; politica LOOK di gestione del disco; politica LRU di rimpiazzamento delle pagine in memoria virtuale. Supponiamo che in un determinato momento ogni pagina logica **n** sia mantenuta su disco nel blocco **n**. Infine supponiamo che i frame di memoria virtuale siano inizialmente non occupati, che i tempi di CPU burst siano trascurabili e che i processi **P1** e **P2** vengano lanciati in parallelo, ove le stringhe di riferimento di P1 e P2 sono le seguenti:

P1: R4, R2, W5, R8, R4

P2: R3, R9, W7, R10, R3

Mostrare l'evoluzione del sistema con una sequenza di colonne simili alla seguente:

contenuto frame 1 e altre info utili (e.g. dirty bit, caricamento in corso)
contenuto frame 2 e altre info utili (e.g. dirty bit, caricamento in corso)
contenuto frame 3 e altre info utili (e.g. dirty bit, caricamento in corso)
contenuto frame 4 e altre info utili (e.g. dirty bit, caricamento in corso)
posizione testina e direzione
coda direzione corrente
coda prossima direzione

Esercizio 2:

a) Fornire un grafo di Holt tale per cui:

- il grafo non contenga un knot
- il grafo sia riducibile a un grafo contenente un knot
- il grafo sia completamente riducibile

b) Mostrare lo stato dell'algoritmo del banchiere che corrisponde al grafo fornito al punto a. Tale stato e' safe?

Esercizio 3: