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 2008/2009
CONCORRENZA – 05 maggio 2010

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

Esercizio 0: Scrivere correttamente nome, cognome, matricola e posizione in tutti i fogli prima di svolgere ogni altro esercizio. Scrivere esclusivamente a penna senza abrasioni. E' vietato l'uso delle penne cancellabili, della matita, dei coprenti bianchi per la correzione (bianchetto) e la scrittura in colore rosso (riservato alla correzione).
Il compito e' formato da due fogli, quattro facciate compresa questa. Le soluzioni che si vuole sottoporre per la correzione devono essere scritte negli spazi bianchi di questi fogli. Non verranno corretti altri supporti.
E' obbligatorio consegnare il compito, e' possibile chiedere che esso non venga valutato scrivendo “NON VALUTARE” in modo ben visibile nella prima facciata.

Esercizio 1: Due nazioni limitrofe che chiameremo 0 e 1 hanno stabilito un numero massimo MAXIMM di immigrati che soggiornano nell'altra nazione.

Quando un cittadino di una nazione vuole andare nell'altra, alla frontiera di rivolge agli ufficiali di immigrazione dell'altra nazione e viene posto in attesa se il numero massimo di immigrati e' stato gia' raggiunto **in una delle due nazioni**.

Pertanto il cittadino di 0 che vuole andare in 1 viene messo in stato di attesa sia se ci sono gia' MAXIMM cittadini di 0 in 1 sia se ci sono MAXIMM cittadini di 1 in 0.

La “vita” dei cittadini e' quindi:

$cittadino_{i,j}, i \in \{0,1\}, j \in 1,\ldots, Ncittadini_i$:

```
while(1) {
    immigration01.emigrate(i); ….
    immigration01.returnhome(i); ….
}
```

Il problema (non la soluzione) presenta problemi di deadlock o di starvation?

Scrivere il monitor **immigration01** con due procedure entry emigrate(int citizenship) e returnhome(int citizenship).

(il parametro e' la propria cittadinanza, I cittadini di 0 mettono 0 come parametro e quindi vogliono andare in 1, e viceversa).

La soluzione chiaramente non deve inserire problemi di deadlock o di starvation che non siano propri del problema.

Esercizio 2: Un semaforo cartesiano planare sghembo prevede due operazioni: Pηβ(x,y) e Vηβ (x,y). Il valore iniziale di tali semafori e' una coppia (xinit, yinit). X, y, xinit e yinit sono numeri reali non negativi.

L'invariante del semaforo cartesiano planare sghembo e':

$$\left(xinit + \sum_{i \in Completed\,V\,\eta\beta} x_i - \sum_{j \in Completed\,P\,\eta\beta} x_j\right)\left(yinit + \sum_{i \in Completed\,V\,\eta\beta} y_i - \sum_{j \in Completed\,P\,\eta\beta} y_j\right) \geq 0$$

I semafori ordinari e quelli cartesiani planari sghembi hanno lo stesso potere espressivo? Produrre una dimostrazione.

Esercizio 3: La seguente classe twostep puo' essere usata al posto della Test&Set?

```
Class twostep {
    private int a;
    private int b;
    void twostep(void) {a=1;b=0}
    atomic int op (int x) {
        int result=b;
        b=a;
        a=x;
        return result;
    }
}
```

Produrre una dimostrazione.

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 2008/2009
PARTE GENERALE – 13 gennaio 2010

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: Il file system di MINIX usa blocchi di 1K byte che vengono numerati con interi (puntatori) a 16 bit. Ogni i-node ha 9 puntatori: 7 puntatori diretti, uno indiretto e uno indiretto-doppio.

1. Qual e' la lunghezza massima dei file in tale file system?
2. A partire da un i-node, indicare quanti blocchi occorre leggere per accedere a un byte in una posizione random nel file. (dalla posizione ??? alla posizione ???, ??? accessi, dalla posizione yyy...)
3. Data L la lunghezza in byte di un file F, scrivere la funzione block_len(L) che associa a L il numero dei blocchi necessari per memorizzare il file F? (il file F e' completamente allocato, non ha “buchi”).

Esercizio 2: Sia dato un algoritmo di minimizzazione delle seek chiamato Odd-Even Look: l'algoritmo soddisfa tutte le richieste pendenti per I cilindri di numero dispari nella direzione crescente, quando non vi sono piu' richieste dispari raggiunge la richiesta con il massimo numero pari e soddisfa durante il percorso di ritorno (descrescente) le richieste relative ai cilindri di numero pari. Quando non vi sono piu' richieste che soddisfano questa condizione l'algoritmo riparte in senso crescente dalla richiesta con il minimo numero dispari di cilindro. Se arriva una richiesta pari quando la testina sta andando in direzione crescente o viceversa dispari quando la direzione e' descrescente, la richiesta viene memorizzata per essere soddisfatta quando verra' invertita la direzione. Se arriva una richiesta dispari durante la scasione in direzione crescente o pari in direzione decrescente ma per cilindri gia' scanditi, la richiesta viene memorizzata e postposta per il giro successivo.

1. Odd-Even Look e' sempre piu' efficiente di C-LOOK (in termini di tempo totale di seek), opuure e' sempre meno efficiente o infine esistono casi in cui e' piu' efficiente e casi in cui e' meno efficiente?
2. E' uniforme? I.e. esistono cilindri scanditi piu' frequentemente di altri?

Documentare appropriatamente la risposta.

Esercizio 3: (obbligatorio) Dato N e' il numero di matricola del candidato, calcolare M=N%5. Dato il capitolo del corso di Sistemi Operativi contrassegnato dal numero M nella lista che segue, indicarne in modo schematico il contenuto e I concetti principali.

0: Gestione della Memoria Principale
1: Scheduling della CPU
2: Gestione delle Risorse
3: File System
4: Gestione della Memoria Secondaria