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 2007/2008
CONCORRENZA – 16 giugno 2009

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 **multiq** che gestisca N code di richieste.

I richiedenti chiamano la funzione che ha la seguente signature:

answer_t multiq.query(int request_type,request_t request);

(request_type e' il numero della coda da 0 a N-1. Esempio di chiamata: result=multiq.query(3,req); nell'esempio N>=4)

Ci sono N gestori, uno per ogni tipo di richiesta. Ogni gestore si comporta come segue:

```
multiq_handler: process[i, i=0,..,N-1] {
   request_t req;
   int type;
   while (1) {
        req=multiq.getquery(i,&type);
        multiq.reply(i,handle(type,req));
   }
}
```

Il gestore i-mo gestisce esclusivamente le richieste di tipo i.

Esercizio 2 (Test&Set): Mostrare se, e nel caso come, le seguenti funzioni possano essere usate per implementare sezioni critiche. (rand(N) genera un numero integer random di valore compreso tra 0 e N-1).

Per le funzioni che possono essere usate per implementare scrivere il protocollo di entrata e di uscita, con commenti che spieghino il principio di funzionamento, per le altre funzioni scrivere chiaramente quale sia il motivo che non consente l'utilizzo delle funzioni per l'implementazione di sezioni critiche.

boh1(int x, int y) = < if (rand(2)) {y=x; x=1;}>

boh2(int x, int y) = < if (rand(2)) {tmp=x; x=y; y=tmp;}>

boh3(int x, int y) = < if (rand(2)) {y=x; x=1;} else {x=y; y=1}>

Esercizio 3: Sia dato un meccanismo di message passing asincrono.

Realizzare una chiamata arecvall() che restituisca tutti i messaggi in attesa per il processo chiamante, concatenandoli.

se non vi sono messaggi in sospeso deve ritornare subito una sequenza vuota (approccio completamente asincrono)

Esercizio 4: (comporta punteggio extra solo se tutti gli altri esercizi hanno valutazione sufficiente. non banale) nelle stesse ipotesi dell'esercizio 1,

Il gestore i-mo gestisce prioritariamente (e non esclusivamente) le richieste di tipo i, ma se non ci sono richieste di tipo i il gestiore i-mo riceve una richiesta della coda con il maggior numero di richieste pendenti (di una delle code con il massimo numero di richieste pendenti qualora ne esista piu' di una con lo stesso numero di richieste).

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 2007/2008
PARTE GENERALE – 16 giugno 2009

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: Un sistema usa un algoritmo di scheduler round-robin con time slice di 3ms.

Il processo P1 viene attivato a t=0, elabora per 5ms, legge un carattere (che arrivera' a t=10) e poi elabora per 4ms.

Il processo P2 viene attivato a t=1, elabora per 3ms, legge un dato da disco (l'accesso richiede 8ms) poi elabora per 7ms.

Il processo P3 viene attivato a t=9, elabora per 10ms, stampa un carattere (la stampante impiega 10ms) poi elabora per 1ms.

1a) Mostrare il diagramma di GANNT del processore.

1b) Mostrare il diagramma di GANNT che si otterrebbe eseguendo i processi su una macchina biprocessore.

Esercizio 2: Sia data la seguente stringa di riferimenti per pagine di 1K per una memoria con 6 frame:

1,2,3,4,5,6,7,8,1,3,5,7,2,4,6,8,1,2,3,4,5,6,7,8

Esercizio 2a) Mostrare il funzionamento dell'algoritmo LRU, calcolare il numero di page fault e la quantita' di dati scambiati col disco (gli accessi sono di sola lettura).

2b) raddoppiando l'ampiezza della pagina i frame diventano 3 e le pagine di indice dispari vengono unificate con le successive. (la pagina 1 e 2 per gli accessi a 1K sono la pagina 1 ampia 2K, la 3 e la 4 sono la pagina 2 della configurazione a 2k e cosi' via).

Mostrare il funzionamento dell'algoritmo LRU, calcolare il numero di page fault e la quantita' di dati scambiati col disco (gli accessi sono di sola lettura). Confrontare il risultato con quello di 2a e verificare per quale ampiezza di pagina vengono scambiati meno dati.

Esercizio 3: Se N e' il numero di matricola calcolare N%5 e spiegare quale e' il funzionamento della system call UNIX corrispondente al risultato. Indicare casi di utilizzo della system call indicata. (non e' necessario ricordare la sintassi)
domanda0: fork
domanda1: execve
domanda2: signal
domanda3: dup
domanda4: wait