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

UNIVERSITA' DEGLI STUDI DI BOLOGNA - CORSO DI LAUREA IN INFORMATICA
PROVA SCRITTA DI SISTEMI OPERATIVI
ANNO ACCADEMICO 2013/2014
16 giugno 2014

Esercizio -1: Essere iscritti su AlmaEsami 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 tre fogli, sei facciate compresa questa. Le soluzioni che si vogliono 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.
Per svolgere questo compito occorre solo una penna e un documento di identità valido. La consultazione o anche solo la disponibilità di altro materiale comporterà l'annullamento del compito (verra' automaticamente valutato gravemente insufficiente).

Esercizio c.1: scrivere il monitor rendezvous che consente ai processi di sincronizzarsi e scambiare dati.

Ogni processo chiama la procedure entry rendezvous.sync specificando due parametri: il numero dei processi con i quali il processo corrente vuole sincronizzarsi e un vettore di ugual numero di interi. Il primo elemento e' inizializzato con il valore conferito dal processo chiamante.

I processi devono coordinarsi con altri che abbiano chiesto di sincronizzarsi con un ugual numero di processi. Quindi, se e quando N processi hanno chiamato la rendezvous.sync chiedendo la sincronizzazione con N processi, tutti N vengono riattivati. Il vettore passato come secondo parametro deve contenere i valori conferiti dagli N processi, uno in ogni elemento del vettore, seguendo l'ordine di chiamata della rendezvous.sync.

es. P1: int v1[]={42,0,0}; rendezvous.sync(3,v1) …. P1 si blocca....

P2: int v[]={314,0,0}; rendezvous.sync(3,v) …. P2 si blocca....

P3: int vv[]={1,0}; rendezvous.sync(2,vv) …. P3 si blocca....

P4: int q[]={13,0,0}; rendezvous.sync(3,q) … P4 sblocca anche P1 e P2. Il secondo paramentro per tutti e tre avra' valore [42,314,13]

….P3 si sblocchera' se e quando un altro processo chiamera' rendezvous.sync con primo parametro 2.

Esercizio c.2: Usando semafori implementare un emulatore di un servizio di message passing asincrono.
Occorre quindi implementare le primitive:
void asend(msg m, pid_t dst)
msg arecv(pid_t sender)
facendo uso di semafori (generali, fifo).
(e' presente una chiamata getpid() che fornisce il pid del chiamante). La asend riceve da un mittente specificato (non e' richiesta la gestione della ricezione da qualunque processo).

Esercizio g.1: Scrivere uno stato per un algoritmo per l'algoritmo del banchiere a tre valute in modo tale che lo stato non sia safe ma se si considerassero solamente due valute alla volta lo stato sarebbe safe.

Esercizio g.2:
a) confrontare gli algoritmi di scheduling Shortest Job Next (SJN) e Round Robin. Quali sono i casi di applicazione dell'uno e dell'altro?
b) quali operazioni compie un file system basato su i-node (per esempio ext2, minix o bffs) quando l'utente digita il comando:
`ln /etc/passwd /tmp`
?
c) quale supporto hardware e' necessario per implementare un algoritmo di rimpiazzamento LRU?