

UNIVERSITA' DEGLI STUDI DI BOLOGNA - CORSO DI LAUREA IN INFORMATICA  
PROVA SCRITTA DI SISTEMI OPERATIVI  
ANNO ACCADEMICO 2013/2014  
24 settembre 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 (verrà automaticamente valutato gravemente insufficiente).

**Esercizio c.1:** Scrivere un monitor vBB che realizzi un bounded buffer a dimensione variabile.

Le procedure entry da implementare sono read, write e resize. Read e write hanno gli stessi parametri del BB ordinario.

La funzione resize ha come parametro un numero intero positivo che rappresenta la nuova dimensione del buffer espressa in numero di elementi. Inizialmente il bounded buffer ha dimensione DEFAULTSIZE. Se la dimensione viene incrementata i processi eventualmente in attesa di scrivere devono poter inserire i loro elementi nel buffer fino a che e' consentito della nuova dimensione.

Se la dimensione diminuisce, nuovi scrittori potranno inserire elementi nel buffer solo quando sarà possibile farlo in modo coerente con il nuovo numero massimo di elementi.

**Esercizio c.2:** dato un servizio di message passing asincrono (FIFO), scrivere un servizio di message passing asincrono LIFO **senza fare uso di processi server**.

Le primitive da implementare sono `lsend(dest, msg)` e `lrecv()`. `lsend` non ha alcun valore di ritorno, `lrecv` restituisce l'ultimo messaggio ricevuto da qualsiasi mittente. Sia il servizio disponibile sia quello da realizzare sono asincroni (`send` mai bloccante, `recv` bloccante in assenza di messaggi, in altre parole **non** sono completamente asincroni)

**Esercizio g.1:** Sia dato questo programma:s

```
Program P:  
  for (i=0;i<2;i++) {  
    long_compute();  
    io_on_dev();  
  }  
  short_compute();
```

`long_compute` impiega 4 ms, `short_compute` 1 ms e `io_on_dev` impiega 2ms. (tutti i processi condividono la stessa risorsa di I/O)  
Considerando in un sistema di elaborazione dove sono in esecuzione tre istanze del programma P che sono state attivate ai tempi 0, 5 e 10ms e che il sistema usa uno shortest remaining time first (SRTF) disegnare il diagramma di Gantt dell'esecuzione. (SRTF e' preemptive). In caso di processi con lo stesso tempo rimanente di esecuzione lo scheduler si comporta come un FCFS (first come first served).

**Esercizio g.2:**

- il supporto DMA e' necessario per implementare un device driver di tipo interrupt driven? Perche'?
- quale supporto hardware e' necessario per l'implementazione di uno scheduler round robin? Perche'?
- quale supporto hardware e' necessario per implementare un algoritmo di rimpiazzamento FIFO? Perche'?