

UNIVERSITA' DEGLI STUDI DI BOLOGNA - CORSO DI LAUREA IN INFORMATICA
 PROVA SCRITTA DI SISTEMI OPERATIVI
 ANNO ACCADEMICO 2012/2013

19 luglio 2013

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 mvie che gestisca M buffer limitati. Ogni buffer ha l'ampiezza di MELEM elementi. I produttori chiamano la procedure entry:

```
put(generic *object)
```

mentre i consumatori chiamano la procedure entry

```
generic *get(int n)
```

I produttori conferiscono un vettore di M elementi, uno per ogni buffer al buffer.

Per esempio put(v), (dove v e' un vettore di M elementi) inserisce ogni elemento del vettore nel buffer corrispondente. I consumatori ricevono un oggetto dal buffer indicato come parametro oggetti ma attendono sempre che ci sia almeno un elemento in ogni buffer.

Esercizio c.2:

```
shared val = 0;
shared Semaphore sp =
new Semaphore(1);
shared Semaphore sq =
new Semaphore(1);
shared Semaphore mutex =
new Semaphore(1);
```

```
process P {
int kp = 3;
while (kp > 0) {
sp.P();
mutex.P();
val = val+1;
sq.V();
mutex.V();
kp--;
}
}
```

```
process Q {
int kq = 2;
while (kq > 0) {
sq.P();
mutex.P();
val = val*2;
sp.V();
mutex.V();
kq--;
}
}
```

a) Al termine di questo programma, quali sono i valori possibili della variabile condivisa val?

b) E' possibile che i processi P o Q restino bloccati indefinitamente?

Esercizio g.1: Siano dati i processi P1, P2 e P3 in ordine decrescente di priorit  (P1 ha massima priorit  e P3 minima).

P1: CPU 2ms, I/O 1ms, CPU 2ms, I/O 2ms, CPU 2ms

P2: CPU 3ms, I/O 2ms, CPU 3ms, I/O 2ms, CPU 3ms

P3: CPU 1ms, I/O 1ms, CPU 1ms, I/O 1ms, CPU 1ms

I tre processi usano la stessa unit  per l'I/O (le richieste di I/O vengono gestite in ordine FIFO)

Si calcoli il diagramma di Gantt (descrivendo il procedimento) per uno scheduler round robin con time slice 2ms.

Esercizio g.2: Le seguenti affermazioni sono vere o false? (Motivare dettagliatamente le risposte).

a) I processi vengono eseguiti pi  velocemente nei sistemi con memoria virtuale.

b) In un File System di tipo FAT non vi e' limite massimo alla dimensione di un singolo file.

c) In un File System di tipo ext2 non vi e' limite massimo alla dimensione di un singolo file.

d) I file system con allocazione contigua non vengono pi  utilizzati.

e) L'algoritmo di rimpiazzamento LRU non puo' essere utilizzato insieme alla tecnica del Working Set.