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 2012/2013

21 giugno 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 (verra' automaticamente valutato gravemente insufficiente).

Esercizio c.1: scrivere un monitor nvie che gestisca N buffer limitati. Ogni buffer ha l'ampiezza di NELEM elementi. I produttori chiamano la procedure entry:

```
put(int n, generic object)
```

mentre i consumatori chiamano la procedure entry

```
get(generic *object)
```

I produttori conferiscono un elemento al buffer indicato come primo parametro. Il secondo parametro e' il valore conferito.

Per esempio `put(3, elem)`, inserisce un elemento nel buffer numero 3.

I consumatori ricevono un vettore di oggetti, al massimo uno per buffer. Devono essere presenti elementi da leggere in almeno N/2 buffer perche' la chiamata abbia successo (negli elementi relativi a buffer vuoti la chiamata restituisce il valore NONE).

Esercizio c.2:

```
shared val = 0;
shared Semaphore sp =
new Semaphore(1);
shared Semaphore sq =
new Semaphore(2);
shared Semaphore mutex =
new Semaphore(1);
```

```
process P {
  int kp = 2;
  while (kp > 0) {
    sp.P();
    mutex.P();
    val = val+1;
    sq.V();
    mutex.V();
    kp--;
  }
}
```

```
process Q {
  int kq = 3;
  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 descrescente di priorita' (P1 ha massima priorita' e P3 minima).
P1: CPU 2ms, I/O 2ms, 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 2ms, CPU 1ms, I/O 2ms, CPU 1ms
I tre processi usano unita' indipendenti per l'I/O (non c'e' contesa per l'I/O)
Si calcoli il diagramma di Gannt (descrivendo il procedimento) per uno scheduler a priorita' di tipo preemptive e lo si confronti con uno scheduler a priorita' di tipo non preemptive.

Esercizio g.2: Si risponda alle seguenti domande:
a) quali sono le differenze fra la paginazione e la segmentazione nella gestione della memoria?
b) quali sono le differenze fra le operazioni P e V dei semafori e le operazioni wait/signal delle variabili di condizione?
c) allocazione contigua, concatenata e indicizzata nei file system, quali sono le differenze e i campi di applicazione?