Nome e cognome _____ N. di matricola (10 cifre) _____ Posizione: Riga ____ Col _____

UNIVERSITA' DEGLI STUDI DI BOLOGNA - CORSO DI LAUREA IN INFORMATICA
CORSO DI SISTEMI OPERATIVI - ANNO ACCADEMICO 2003/2004
MIDTERM CONCORRENZA - 14 Novembre 2003

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:

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

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

```
process Q {
  int kq = 3;
  while (kq > 0) {
    sq.P();
    mutex.P();
    val = val*2;
    sp.V();
    kq--;
    mutex.v();
  }
}
```

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 2: Buffer di accumulazione doppio

Si consideri il seguente problema. Esistono due classi di oggetti chiamati A e B. Gli oggetti di queste due classi vengono prodotti da un insieme di produttori P_{AB} e consumati da due insiemi di consumatori C_A e C_B. I produttori producono un oggetto di classe A e un oggetto di classe B con una singola chiamata al monitor DoubleBuf. I consumatori in C_A consumano oggetti di classe A e i consumatori in C_B consumano oggetti di classe B. Gli oggetti vengono mantenuti in due buffer limitati (entrambi di dimensione N, con N pari) controllati dal monitor DoubleBuf. Vi sono numerosi produttori/consumatori (il numero di istanze di ogni categoria supera N).

Il monitor è caratterizzato da due periodi: riempimento e svuotamento. Durante il riempimento è possibile solo aggiungere oggetti ai buffer, durante lo svuotamento è possibile solo rimuovere oggetti. Quando il buffer è vuoto, i produttori inseriscono oggetti fino a riempirlo. Quando il buffer è pieno, i consumatori consumano oggetti fino a svuotarlo. I consumatori devono alternarsi nel modo seguente: prima viene consumato un oggetto A, poi viene consumato un oggetto B, poi ancora un oggetto B, poi un oggetto A, e così via fino allo svuotamento del buffer.

```
process P_AB {
  while (true) {
    A a = produceA();
    B b = produceB();
    DoubleBuf.add(a, b);
  }
}
```

```
process C_A {
  while (true) {
    A a = DoubleBuf.getA();
    consume(a);
  }
}
```

```
process C_B {
  while (true) {
    B b = DoubleBuf.getB();
    consume(b);
  }
}
```

a) Scrivere il monitor DoubleBuf
b) Sono possibili situazioni di deadlock?
c) Sono possibili situazioni di starvation?

Esercizio 3 – Buffer con messaggi, variazione
Sia dato un buffer con N elementi e siano dati numerosi produttori/consumatori (il numero di istanze di ogni categoria supera N). Si realizzi un meccanismo per la comunicazione di dati dai produttori ai consumatori che rispetti tutte le proprietà del buffer limitato ma consenta ad un consumatore di prelevare il dato solamente se il buffer è completamentente pieno oppure se ci sono in attesa tanti conumatori quanti elementi nel buffer (N).
Scrivere le funzioni bbbsend e bbbreceive (blocking bounded buffer send/receive), facendo uso di message passing sincrono e di un processo gestore.