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

UNIVERSITA' DEGLI STUDI DI BOLOGNA - CORSO DI LAUREA IN INFORMATICA
PROVA **PARZIALE** DI SISTEMI OPERATIVI
ANNO ACCADEMICO 2023/2024
23 gennaio 2024

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 due fogli, quattro 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 il monitor `ab12` che gestisce due buffer limitati FIFO, il primo memorizza elementi di tipo `tipoa` il secondo elementi di tipo `tipob`. Entrambi i buffer contengono al massimo MAX elementi.
Il monitor `ab12` fornisce quattro procedure entry:
`void adda(tipoa a1, tipoa a2):` aggiunge due elementi al primo buffer.
`void addb(tipob b1):` aggiunge un elemento al secondo buffer.
`tipoa getone(void):` restituisce un elemento dal primo buffer.
`(tipoa, tipob) getab(void)`: restituisce un elemento dal primo buffer e uno dal secondo.
Tute le chiamate possono essere bloccanti (se non c'è spazio per inserire elementi nel caso delle chiamate di tipo `add`, se non ci sono gli elementi da restituire per le chiamate di tipo `get`). Le richieste devono essere soddisfatte in ordine FIFO.

Esercizio c.2:
Si consideri il programma che ha quattro variabili globali:

```
semaphore s1(1);
semaphore s2(1);
semaphore mutex(1)
int n=0;
```

e due thread:

```
Thread A {
  register int i;
  for (i=1;i<3;i++) {
    s1.P();
    mutex.P()
    n+=i;
    mutex.V()
    s2.V();
  }
}
```

```
Thread B{
  register int j;
  for (j=2;j<4;j++) {
    s2.P();
    mutex.P()
    n*=j;
    mutex.V()
    s1.V();
  }
}
```

Mostrare tutti i possibili valori di n al termine dell'esecuzione, corredando la risposta con opportuna spiegazione del ragionamento seguito per trovare Il risultato.

PRECISAZIONE:
nell'esercizio C1 la frase "Le richieste devono essere soddisfatte in ordine FIFO" deve intendersi:

- Le richieste di tipo add (adda, addb) devono essere soddisfatte in ordine FIFO.
- Le richieste di tipo get (getone, getab) devono essere soddisfatte in ordine FIFO.

Es. Se un processo P chiama la adda e si blocca perché non c'è spazio sufficiente nel primo buffer, il processo Q che successivamente chiama addb dovrà attendere anche se ci sarebbe spazio nel secondo buffer.
In nessun caso Q deve poter completare che richiesta prima di P.
Ugualmente se c'è un processo in attesa di fare la getone o la getab, successive chiamate getone o getab devono aspettare il proprio turno.