UNIVERSITÀ DEGLI STUDI DI BOLOGNA – CORSO DI LAUREA IN INFORMATICA
CORSO DI SISTEMI OPERATIVI
PRIMA PROVA PARZIALE – ANNO ACCADEMICO 2002/2003
29 ottobre 2002

Esercizio -1: essersi iscritti correttamente per svolgere questa prova.
Esercizio 0: Scrivere correttamente il proprio nome, cognome e numero di matricola in ogni foglio prima di svolgere ogni altro esercizio seguente.
Esercizio 1: Sia dato un sistema dove oltre alle azioni atomiche di lettura e scrittura in memoria sia data la seguente operazione atomica che opera su due variabili di tipo intero:

```
f(a,b)=<a=b=(b+3)%12>
```

Scrivere, se possibile, un meccanismo di supporto di sezioni critiche in modo simile al test&set usando la funzione f; in caso contrario fornire una spiegazione del perché non sia possibile scrivere tale supporto con la funzione f.

Esercizio 2: Il seguente programma concorrente, a seconda dell'interleaving fra i due processi, puo' stampare una o piu' stringhe. Indicare quali. (nota: tutti i semafori sono semafori generali inizializzati a 0)

```
P: process {
  s1.P();
  print A;
  s1.P();
  s2.V();
  print C;
}
```

```
Q: process {
  print B;
  s1.V();
  s1.V();
  s2.P();
  print D;
}
```

Esercizio 3: Considerate i seguenti processi:

```
P: process {
  print C;
  print I;
  print O;
}
```

```
Q: process {
  print A;
  print N;
}
```

Modificare il codice dei processi in modo che stampino unicamente le parole "CIANO" e "CAINO". Le uniche modifiche ammesse sono inserimenti di invocazioni di primitive P e V sui semafori definiti da voi. Tutti i semafori utilizzati devono essere inizializzati a zero.

Esercizio 4: In un museo di arte moderna, esistono due tipi di utenti: le persone singole che visitano il museo da soli, e le persone che fanno parte di gruppi organizzati. Le persone di un gruppo visitano il museo tutti insieme e usufruiscono di particolari sconti.
Il museo e' piccolo, e ha una capacita' limitata $CMAX$ (sufficiente pero' ad accogliere il piu' grande dei gruppi). Se il museo e' pieno, gli utenti (singoli o di gruppo) si mettono in fila per entrare.
La visita di una persona singola e di un partecipante a un gruppo si svolgono rispettivamente nei seguenti modi:

```
utente-singolo: process {
  museo.entraPersonaSingola();
  // ammira opere
  museo.esciPersonaSingola();
  // vai allo shop per comprare un poster
}
```

```
utente-gruppo: process {
  gruppo = miogruppo()
  museo.formaGruppo(gruppo);
  museo.entraPersonaInGruppo(gruppo);
  // ammira opere
  museo.esciPersonaInGruppo(gruppo);
  // vai allo shop per comprare un poster
}
```

Valgono le seguenti regole:

- i visitatori di un gruppo organizzato si devono prima raccogliere in gruppo all'entrata del museo
- quando il gruppo e' formato, i membri entrano nel museo uno alla volta, uno dopo l'altro
- mentre un gruppo sta entrando, nessuno altro utente (singolo o di gruppi diversi) puo' entrare (per evitare che si intrufoli nel gruppo)
- le persone in gruppo, una volta dentro, si comportano come persone singole; ovvero visitano il museo e escono quando ne hanno voglia

Il museo si e' organizzato con un sistema di prenotazione per gruppi, per cui si garantisce che ad ogni istante ci possono essere al piu' $GMAX$ gruppi in attesa fuori dal museo, identificati dagli indici $0..GMAX-1$. Quando un gruppo e' entrato, il suo indice ritorna libero e un altro gruppo si potra' formare con lo stesso indice. Per evitare confusione, pero', fino a quando esistono membri di un gruppo di indice i all'interno del museo, un gruppo con lo stesso indice che aspetta fuori non puo' entrare. Solo quando tutti i membri del gruppo precedente sono usciti possono entrare. La funzione $dimensione(inti)$ ritorna la dimensione del gruppo i.

- A) La definizione data non e' starvation-free. Definire un invariante che garantisca assenza di starvation.
- B) Data la definizione del problema, possibile che vi siano deadlock?
- C) Implementare il monitor museo