Università di Napoli Federico II
Corso di Laurea in Ingegneria Informatica e Elettronica
Esame di Sistemi Operativi
Proff. De Carlini, Cotroneo, Cinque

Prova pratica del 13/09/2012
Durata della prova: 150 minuti

***Cognome** ....…….................. **Nome** ...…............................... **Matr.** ....*

°°

*Lo studente legga attentamente il testo e produca il programma, il makefile, ed i casi di test necessari per dimostrarne il funzionamento. La mancata compilazione dell'elaborato, la compilazione con errori o l'esecuzione errata del programma daranno luogo alla valutazione come **prova non superata**. Ricordarsi di indicare Nome, Cognome e matricola su questo stesso foglio, che dovrà essere in ogni caso consegnato alla Commissione. Al termine della prova lo studente dovrà fare verificare il funzionamento del programma ad un membro della Commissione.*

°°

Testo della prova

Si implementi in linguaggio C/C++ lo schema **produttore-consumatore a priorità** illustrato di seguito. Occorre che vi siano 2 tipi di processi produttori, indicati rispettivamente come **produttori ad alta priorità** e **produttori a bassa priorità**. Come nel classico problema del produttore-consumatore, i produttori di entrambi i tipi non possono produrre finché non vi è un buffer disponibile. Inoltre, è necessario implementare il seguente vincolo: **un produttore a bassa priorità può produrre solo se non vi è alcun produttore ad alta priorità in attesa di un buffer libero**. Lo schema deve essere basato sul costrutto **monitor** (con semantica **signal-and-continue**), introducendo un opportuno numero di *condition variables* per gestire i vincoli del problema. Si consideri il caso di un **vettore circolare di buffer di tipo intero**, gestito con due puntatori logici *testa* e *coda*, e i seguenti prototipi:

```
typedef struct {
      int buffer[3];
      int testa; int coda;
      Monitor m;           // utilizzare la libreria di procedure allegate
} PriorityProdCons;[1]

void inizializza_prod_cons(PriorityProdCons * p);
void produci_alta_prio(PriorityProdCons * p);
void produci_bassa_prio(PriorityProdCons * p);
void consuma(PriorityProdCons * p);
void rimuovi_prod_cons(PriorityProdCons * p);
```

Il programma dovrà istanziare 1 processo produttore ad alta priorità, 3 processi produttori a bassa priorità, ed 1 processo consumatore. Il processo produttore ad alta priorità invoca per 3 volte il metodo `produci_alta_prio`, il quale produce un valore casuale tra 0 e 12, attendendo 2 secondi tra una invocazione e la successiva. I processi produttori a bassa priorità invocano per 3 volte il metodo `produci_bassa_prio`, il quale produce un valore casuale tra 13 e 25, attendendo 1 secondo tra le invocazioni. Il processo consumatore consuma e stampa a video un elemento invocando `consuma` per 12 volte, e attendendo 1 secondo tra le invocazioni. Il programma principale attende la terminazione dei processi figli per poi terminare a sua volta.

---

[1] Si allochi `PriorityProdCons` su una memoria condivisa UNIX; si utilizzi la sintassi `&(p->m)` per ottenere un puntatore alla variabile `Monitor` da utilizzare nelle procedure allegate.