Università degli Studi di Napoli Federico II
Corso di Laurea in Ingegneria Informatica
Esame di Sistemi Operativi
Proff. Cotroneo, Natella

Prova pratica del 16/10/2014
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 realizzi in linguaggio C/C++ una applicazione **multithread**, basata sul costrutto **monitor**, in cui siano previsti tre tipi di thread: un thread **generatore**, un thread **aggiornatore**, e 3 thread di tipo **destinatario**. I thread dovranno cooperare nella seguente maniera:

- Il thread **generatore** dovrà (per 10 volte) generare un elemento (una struttura contenente una coppia di interi casuali tra 0 e 10), e inserirla nella coda di un vettore circolare di buffer, usando il metodo ***genera()***. Se il vettore è pieno, il thread deve sospendersi finché non si rende libero un buffer del vettore.
- Il thread **aggiornatore** dovrà periodicamente (ogni secondo, 10 volte) prelevare un elemento dalla testa del vettore con il metodo ***preleva()***. Se il vettore è vuoto, il thread deve attendere finché non si rende disponibile un elemento.
- Dopo aver prelevato un elemento, e prima di prelevarne uno ulteriore, il thread **aggiornatore** dovrà copiare l'elemento all'interno di una struttura dati condivisa con i thread **destinatario**, utilizzando il metodo ***aggiorna()***. Il thread aggiornatore dovrà sospendersi nel caso vi siano thread destinatari che stiano consultando il buffer.
- I thread **destinatari** dovranno periodicamente (ogni 2 secondi, per 6 volte) consultare il valore corrente della struttura, usando il metodo ***consulta()***, che stamperà a video i singoli valori interi e la loro somma. Se il thread aggiornatore sta modificando la struttura dati, i thread destinatari dovranno sospendersi finché la modifica non sarà terminata.

Per gestire le strutture dati, si realizzino due strutture monitor con i seguenti metodi:

```
typedef struct { int a; int b; } elemento;
typedef struct {
      elemento vettore[5];
      // inserire qui variabili per la sincronizzazione
} MonitorVettore;
typedef struct {
      elemento buffer;
      // inserire qui variabili per la sincronizzazione
} MonitorBuffer;
void genera(MonitorVettore *v, elemento e);
elemento preleva(MonitorVettore *v);
void aggiorna(MonitorBuffer *b, elemento e);
void consulta(MonitorBuffer *b);
```