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 07/07/2011
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++ un tipo di dato astratto **Stack** che sia **thread-safe**, ossia i cui metodi siano richiamabili da più thread garantendo che non più di un thread alla volta possa modificare i dati incapsulati. Si utilizzi il costrutto **Monitor** basandosi sulla libreria **PThreads**. Lo Stack deve realizzare la seguente interfaccia (oppure, a scelta dello studente, il suo equivalente C++ usando una classe):

```
typedef int Elem;

typedef struct Stack {
      Elem * dati;      // puntatore ad un vettore dinamico di elementi
      int dim;          // dimensione dello Stack
      ...               // variabili da aggiungere per la sincronizzazione
} Stack;

void StackInit(Stack * s, int dim); // alloca dinamicamente "dim" elementi
void StackRemove(Stack * s);        // dealloca gli elementi
void StackPush(Stack * s, Elem e);  // inserimento nello Stack
Elem StackPop(Stack * s);           // estrazione dallo Stack
int StackSize(Stack * s);           // ritorna il numero attuale di elementi
```

Se un thread invoca un metodo dello Stack mentre un altro thread sta già eseguendo un metodo, esso deve essere messo in attesa finché l'altro thread non ha terminato l'esecuzione del metodo. Se lo Stack è vuoto, un thread che invoca `StackPop()` deve essere messo in attesa fino a quando un nuovo elemento sarà stato inserito. Se lo Stack è pieno, un thread che invoca `StackPush()` deve essere messo in attesa fino a quando un elemento sarà stato rimosso.

Si sviluppi inoltre un programma che utilizzi lo Stack. Il programma deve istanziare 5 thread, ciascuno dei quali inserisce un valore intero casuale tra 0 e 10 nello Stack e attende un secondo prima di ripetere l'inserimento; ogni thread ripete l'operazione per 4 volte. Il programma deve inoltre istanziare un thread che ripetutamente estrae 2 valori dallo Stack e ne stampa la somma a video; l'operazione viene ripetuta 10 volte. Una volta generati i thread (come joinable), il programma principale ne attende la terminazione, e termina a sua volta.