

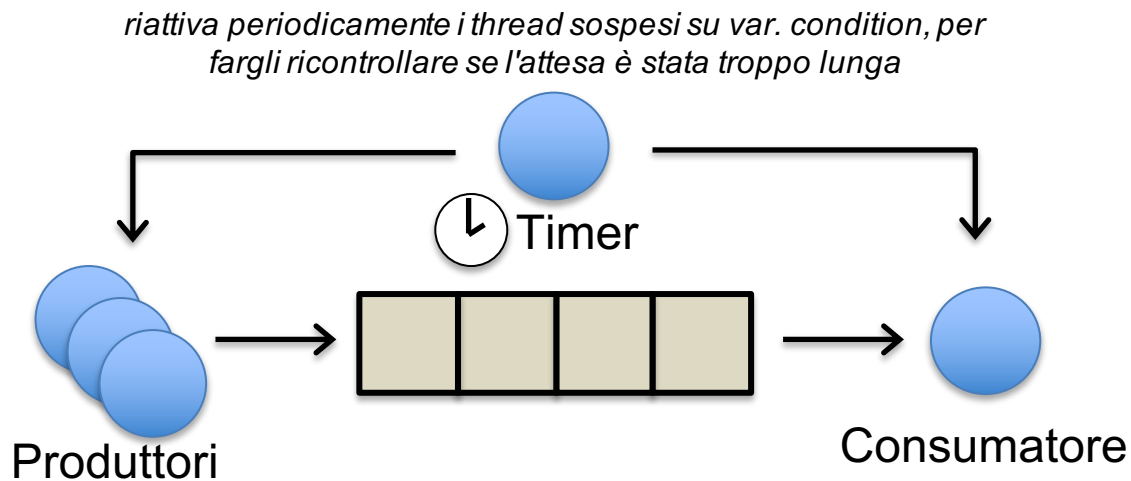
Testo della prova

Si realizzi in linguaggio C/C++ un'applicazione **multithread** basata sul **costrutto Monitor**, che realizzi lo schema **produttore-consumatore** con un vettore di buffer circolare.

In aggiunta allo schema di sincronizzazione classica, si richiede la seguente **variante**. Si supponga di voler impedire che i thread rimangano sospesi in attesa per un **tempo eccessivo**. Il metodo *"inizializza_monitor"*, oltre ad inizializzare la struttura dati, dovrà avviare un thread aggiuntivo che funga da "timer". Il thread timer esegue un ciclo, in cui periodicamente (ogni secondo, attendendo con *sleep()*) effettui una *signal_condition* su tutti i produttori e su tutti i consumatori. Il thread timer termina quando la funzione *"distruggi_monitor"* pone ad 1 una variabile *"uscita_timer"*, da dichiarare all'interno del monitor.

I thread produttori e consumatori si pongono normalmente in attesa che il buffer non sia già pieno o vuoto. Quando i thread sono riattivati da *signal_condition*, prima di controllare se la condizione è verificata, dovranno controllare se è trascorso un tempo superiore a 3 secondi da quando sono entrati nel monitor. Se il tempo di attesa è troppo elevato, il metodo esce dal monitor senza porsi nuovamente in attesa sulla condition variable, e ritorna in uscita un valore negativo.

Simulare l'esecuzione di 5 thread produttori e 1 thread consumatore. Ogni produttore inserisce produce 4 valori (scelti casualmente) in un ciclo, senza porsi in attesa tra le iterazioni. Il consumatore effettua 20 consumazioni in un ciclo, attendendo 2 secondi tra le iterazioni. Il produttore dovrà verificare il valore di ritorno del metodo di produzione, e se tale valore è negativo, si limiterà a ritentare la produzione dopo 1 secondo di ulteriore attesa.



File da completare:

- main.c
- prodcons.h
- prodcons.c