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

Prova pratica del 18/01/2021 - TURNO 1
Durata della prova: 75 minuti

ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo

Lo studente completi il programma a corredo di questo documento, seguendo le seguenti indicazioni.

La prova sarà valutata come segue:

- **A**: Prova svolta correttamente.
- **B**: Il programma non esegue correttamente, con errori minori di programmazione o di concorrenza.
- **C**: Il programma non esegue correttamente, con errori significativi (voto max: 22).
- **INSUFFICIENTE**: Il programma non compila o non esegue, con errori gravi di sincronizzazione.

Istruzioni per la consegna dell'elaborato

L'elaborato dovrà essere svolto in una cartella dal nome: `Cognome_Nome_Matricola_Docente`

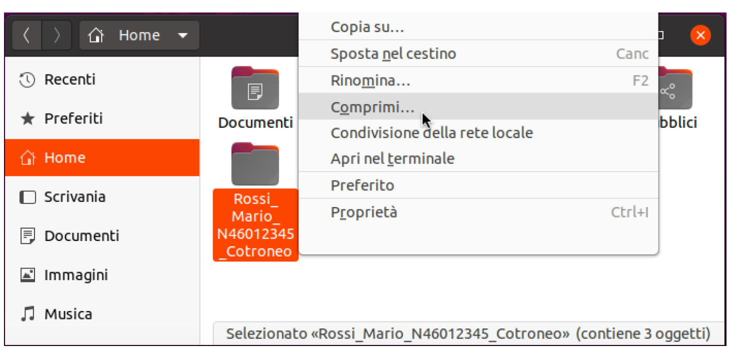
Esempio:

```
mkdir Rossi_Mario_N46012345_Cotroneo
cd Rossi_Mario_N46012345_Cotroneo
```

…. copiare nella cartella i file forniti per l'esame ….
…. sviluppare il programma ….
…. per la consegna, dalla shell (assumendo di essere ancora nella cartella di lavoro), creare un file compresso ("tar") con i seguenti comandi:

```
cd ..
tar -czvf ./Rossi_Mario_N46012345_Cotroneo.tar.gz  ./Rossi_Mario_N46012345_Cotroneo
```

In alternativa, è consentito creare il file compresso "tar" tramite l'interfaccia grafica.



All'interno della macchina virtuale, aprire il browser all'indirizzo: **https://tinyurl.com/y97qfmx9**

Nel form, cliccare su "**aggiungi file**", selezionare il file compresso contenente il proprio svolgimento, e indicare il proprio nome ed email @studenti.unina.it.

Attendere una notifica del docente, e quindi scollegarsi dalla piattaforma di VirtualClassroom.

ooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo

Testo della prova

Si realizzi in linguaggio C/C++ un processo servente **multithread** basato su **code di messaggi UNIX**. Il processo servente, denominato **Server**, riceve misure provenienti da un gruppo di 3 processi denominati **Client** che rappresentano dei sensori. Ogni client invia 10 misure in maniera asincrona su **un'unica coda**. Ogni messaggio inviato dovrà contenere un identificativo intero del Client (tra 1 e 3) e il valore della misura (intero), selezionato casualmente tra 0 e 100 con la funzione rand(). Una volta inviate le 10 misure, i processi Client terminano. Il codice dei Client e del Server deve risiedere in **due eseguibili distinti**, che sono invocati da un **programma principale** attraverso una delle varianti della primitiva **exec**. Quando tutti i Client e il Server terminano, il programma principale rimuove la coda e termina a sua volta.

Il Server istanzia 2 tipi di thread, rispettivamente 3 **Worker** e 1 **Collector**. I thread interagiscono tramite un Buffer condiviso, su cui sincronizzare l'accesso mediante il costrutto monitor, in base ai seguenti vincoli di sincronizzazione.

Ogni worker è dedicato alla ricezione (bloccante) selettiva dei messaggi di uno specifico client (worker 1 per client 1, e così via). Ogni worker effettua ed elabora 10 ricezioni e poi termina. Alla ricezione di un messaggio, il worker chiama il metodo "*aggiorna*" del monitor, che incrementa la variabile *somma* contenuta nel Buffer, aggiungendo al valore corrente di *somma* il valore di misura ricevuto dal client. Il metodo "aggiorna" si limita alla mutua esclusione tra thread, senza porre i worker in attesa di alcuna condizione di sincronizzazione.

Il thread collector chiama il metodo "*preleva*" del monitor, che lo pone in attesa della condizione che siano stati ricevuti 30 messaggi (10 ricezioni x 3 client). Quando si verifica questa condizione, il thread collector riprende ad eseguire. Esso preleva la somma finale (che nel frattempo avrà accumulato 30 aggiornamenti), calcola la media delle misure (dividendo la somma finale per 30), stampa il risultato a video, e termina.