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

Prova pratica del 16/6/2015
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 programma basato su **code di messaggi UNIX**, che generi tre processi **Client** ed un processo **Server**. I Client dovranno inviare messaggi al Server con una coda condivisa attraverso una **send sincrona**, da implementare a partire dalle primitive *msgsnd()* e *msgrcv()*. Il processo Server dovrà istanziare **due thread** che, parallelamente, dovranno entrambi prelevare messaggi di richiesta dalla coda condivisa. Ad esempio, entrambi i thread dovranno effettuare delle *receive* sulla coda condivisa; il thread che preleverà un messaggio in arrivo è scelto a discrezione del sistema operativo.

Quando un thread riceve una "request to send", esso dovrà rispondere con una "ok to send" al Client che ha generato la richiesta; lo stesso thread dovrà poi prelevare ed elaborare il messaggio che il Client invierà a seguito della "ok to send". **Entrambi i thread dovranno usare le stesse code condivise**. Per evitare interferenze fra i thread, si includa all'interno di ogni messaggio il **PID del Client** (sfruttando il campo "tipo" del messaggio), e **due valori interi** tra 0 e 10, scelti casualmente. I messaggi contenenti uno stesso PID verranno gestiti dallo stesso thread. Alla ricezione di un messaggio, il thread dovrà stampare il PID del Client che ha generato il messaggio e la somma dei due valori interi.

Ogni Client dovrà generare 4 messaggi e attendere le due corrispondenti risposte, per poi terminare. Ciascuno dei due thread del Server dovrà elaborare 6 messaggi, per poi terminare. Il processo Server dovrà terminare quando entrambi i suoi thread hanno terminato. Il codice dei Client e del Server deve risiedere in **due eseguibili distinti**. Si crei inoltre un terzo eseguibile che avvii gli altri due.