Esercitazione: code di messaggi

Esercizio 1: esplorazione della struttura msqid_ds

Scrivere un programma in cui è creata una coda di messaggi.

1. Prima di effettuare qualsiasi operazione di invio di messaggi sulla coda, utilizzando la struttura di tipo msqid_ds associata alla coda, stampare la dimensione della coda (membro msg_qbytes), il numero di messaggi in coda (membro msg_qnum) e il tempo dell'ultima msgsnd() (deve essere 0, poiché non ci sono ancora stati invii sulla coda).
2. Effettuare una msgsnd(), e verificare che il momento della msgsnd() è cambiato. Utilizzare la funzione ctime() di time.h per visualizzare il momento in cui è occorsa la msgsnd().
3. Prima di terminare il programma deve deallocare la coda.

Esercizio 2: costruzione di interfaccia alle system call msgsnd() e msgrcv()

Scrivere 2 programmi che funzionino da interfaccia alla msgsnd() e alla msgrcv().

1. Il primo, intf_snd.c (che fornisce un'interfaccia alla msgsnd()) deve creare una coda di messaggi: dopo la creazione della coda, entra in un ciclo di richiesta all'utente in cui a ogni interazione richiede all'utente di inserire un tipo di messaggio (un intero!) e una stringa. Tipo di messaggio e stringa saranno inseriti in un messaggio, e il messaggio inviato sulla coda. Scegliere una sequenza di caratteri che permetta di interrompere tale ciclo e terminare.
2. Il secondo programma, intf_rcv.c (che fornisce un'interfaccia alla msgrcv()) deve connettersi alla stessa coda creata dal primo, domandare all'utente che tipo di messaggi desideri prelevare dalla coda, prelevarli e stamparli. Al termine della stampa dei messaggi, il processo deve rimuovere la coda.

Esercizio 3: invio e ricezione messaggi

Scrivere un programma che allochi una coda di messaggi tramite msgget(), generi un processo figlio e si metta quindi in attesa di un messaggio inviato dal figlio.

Il processo figlio deve collegarsi alla coda generata dal processo padre e inviare su di essa un messaggio costituito da due dati: una stringa (la stringa "saluti da") e il PID del processo figlio. Fatto questo il processo figlio termina. Ricevuto il messaggio atteso, il processo padre visualizza a video quando ricevuto, dealloca la coda e termina.

Esercizio 4: rimozione delle code

Verificare che la coda è stata deallocata usando il comando ipcs da terminale. Se non risulta deallocata, rimuoverla utilizzando ipcrm.

Esercizio 5:

Scrivere un programma nel quale un processo padre intende comunicare con NUM_KIDS processi figli attraverso le code di messaggi, secondo lo schema:

1. Ogni figlio estrae un numero intero positivo casuale minore o uguale a NUM_MAX
2. Ogni figlio invia al padre attraverso una coda di messaggi il numero estratto e poi si mette in attesa (scegliere il meccanismo più opportuno)
3. Il padre legge tutti i numeri inviati dai figli e ne estrae il minimo
4. Il padre comunica al processo figlio che ha inviato il valore minimo di terminare la sua esecuzione (nel caso di numero minimo ex-equo, si scelga un processo con una strategia a piacere)
5. Il padre comunica a tutti gli altri figli di estrarre un altro numero casuale e di spedirlo.
6. Il procedimento termina quando tutti i figli sono terminati
7. Nota: **se necessario**, invece della system call pause(), si consiglia di usare la system call sigsuspend() – Vedere i dettagli della call nelle pagine del manuale