SISTEMI OPERATIVI
(A.A. 99-00)
16 GIUGNO 2000

IMPORTANTE:

1) I due esercizi possono essere svolti durante due prove scritte differenti. L'indicazione di quale esercizio si vuole svolgere viene ricavato dalla lista d'esame: l'assenza di indicazioni corrisponde implicitamente alla volontà di svolgere entrambi gli esercizi. Si considerano parte integrante delle soluzioni i **COMMENTI *significativi*** introdotti per facilitare la lettura del codice: come tali, essi influenzano la votazione finale.
2) La soluzione dell'esercizio 1 deve essere riportata sui fogli a quadretti, mentre la soluzione dell'esercizio 2 deve essere riportata sui fogli bianchi scrivendo SOLO SU UN LATO: su ogni foglio indicare nome, cognome, numero di matricola e numero di esercizio e **numero del compito**.
3) La soluzione all'esercizio n. 2 verrà **fotocopiata** al termine dell'esame scritto. Lo studente dovrà produrre un insieme di file corrispondenti alla soluzione completa e il relativo listato, che verranno analizzati durante la prova pratica. Inoltre, lo studente dovrà segnalare le variazioni rispetto alla versione fotocopiata prodotta durante l'esame scritto (le eventuali parti eliminate vanno riportate sotto forma di commento).

ESERCIZIO N. 1

Una piccola biblioteca ha a disposizione **L** libri e **P** (< **L**) posti a sedere per la consultazione. Ogni utente richiede 3 libri, che consulta contemporaneamente; gli utenti prendono posto a sedere solo se sono disponibili tutti i libri che hanno richiesto e c'è posto a sedere. Terminata la consultazione, ogni utente deve restituire tutti i libri che ha avuto in prestito.
Si implementi una soluzione usando il costrutto monitor per modellare la biblioteca e i processi per modellare gli utenti, e si descriva la sincronizzazione tra gli utenti. Nella soluzione si massimizzi l'utilizzo delle risorse. Si discuta se la soluzione proposta può presentare starvation e in caso positivo per quali processi, e si propongano modifiche e/o aggiunte per evitare starvation.

ESERCIZIO N. 2

Si realizzi un programma **concorrente** per UNIX che deve avere una parte in **Bourne Shell** e una parte in **C**. La parte in Shell deve prevedere due parametri: il primo parametro deve essere il **nome assoluto di un direttorio** che identifica una gerarchia (**G**) all'interno del file system; il secondo parametro deve essere una **stringa** (**S**). Il programma deve cercare nella gerarchia **G** specificata tutti i direttori che contengono almeno un file in cui la stringa **S** sia presente sia nel *nome* sia nel *contenuto*. Per ogni direttorio trovato, si deve riportare il nome assoluto sullo standard output e si deve invocare la parte C passando come parametri i nomi dei file che soddisfano la condizione.
La parte in C accetta un numero variabile di parametri che rappresentano nomi di file e deve creare un numero di figli pari al numero di file passati come parametri. I figli devono eseguire in modo concorrente in modo da massimizzare il parallelismo. Ogni processo figlio cerca nel file corrispondente se ci sono caratteri (chiamati **CE** nel seguito) con un codice ASCII maggiore di 127. Per ogni carattere trovato, il figlio deve notificare il padre il quale tiene conto di quante notifiche ha ricevuto. Quando sono stati trovati complessivamente più di 20 **CE**, il padre comunica a tutti i figli di terminare l'esecuzione. In ogni caso (terminazione prematura o perché il file è finito) ogni figlio deve comunicare al padre quanti **CE** sono stati trovati. Il padre deve riportare sullo standard output la differenza tra le notifiche ricevute e la somma dei valori comunicati dai figli a fine esecuzione.
Si discutano i vantaggi e gli svantaggi del metodo scelto per la comunicazione tra i processi. Inoltre si discuta se nella soluzione proposta esistono dipendenze dai tempi di esecuzione, e si propongano soluzioni per eliminarle.