SISTEMI OPERATIVI e
LABORATORIO DI SISTEMI OPERATIVI
(A.A. 12-13) – 12 FEBBRAIO 2014

IMPORTANTE:

1) Fare il login sui sistemi in modalità Linux usando il proprio **username** e **password**, attivare syncexam.sh e passare in modalità testuale.
2) I file prodotti devono essere collocati in un **sottodirettorio** (che deve essere nella directory studente_XXX) che deve essere creato e avere nome **ESAME12Feb14_1_01.** FARE ATTENZIONE AL NOME DEL DIRETTORIO, in particolare alle maiuscole e ai trattini indicati. Verrà penalizzata l'assenza del direttorio con il nome indicato e/o l'assenza dei file nel direttorio specificato, al momento della copia automatica del direttorio e dei file. **ALLA SCADENZA DEL TEMPO A DISPOSIZIONE VERRÀ INFATTI ATTIVATA UNA PROCEDURA AUTOMATICA DI COPIA, PER OGNI STUDENTE DEL TURNO, DEI FILE CONTENUTI NEL DIRETTORIO SPECIFICATO.**
3) Il tempo a disposizione per la prova è di **75 MINUTI** per lo svolgimento della sola parte C e di **120 MINUTI** per lo svolgimento di tutto il compito.
4) Non è ammesso **nessun tipo di scambio di informazioni** né verbale né elettronico, pena la invalidazione della verifica.
5) L'assenza di commenti significativi verrà penalizzata.
6) **AL TERMINE DELLA PROVA È INDISPENSABILE CONSEGNARE IL TESTO DEL COMPITO (ANCHE IN CASO CHE UNO STUDENTE SI RITIRI): IN CASO CONTRARIO, NON POTRÀ ESSERE EFFETTUATA LA CORREZIONE DEL COMPITO MANCANDO IL TESTO DI RIFERIMENTO.**

Esercizio

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 che devono essere nomi assoluti di direttori che identificano due gerarchie (**G1** e **G2**) all'interno del file system. Il programma deve cercare (in due fasi successive) nelle gerarchie **Gi** specificate (prima **G1** e poi **G2**) tutti i direttori che contengono almeno **un** file la cui lunghezza in byte sia **pari** e strettamente minore di **510**: si riporti il nome assoluto di tali direttori sullo standard output. **Al termine dell'intera esplorazione ricorsiva di G1 e di G2**, si deve invocare la parte in C passando come parametri i nomi assoluti di tutti i file trovati (**F1**, **F2**, ... **FN-1, FN**), avendo cura di passare **prima quelli trovati in G2 e poi quelli trovati in G1**.

La parte in C accetta un numero variabile **N** di parametri maggiore o uguale a 2 (*da controllare che N sia ≥ 2*) che rappresentano i nomi assoluti di file **F1**, **F2**, ... **FN-1, FN** *(tutti con lunghezza minore di 510, da non controllare)*. Il processo padre deve generare **N processi figli** (**P0 … PN-1**) e ognuno dei processi figli deve generare un processo nipote (**PP0 … PPN-1**): ogni processo figlio **Pj** e il suo processo nipote **PPj** costituiscono una coppia che è associata al corrispondente file **Fj+1**. Ogni processo figlio **Pj** deve, prima di creare il proprio nipote, creare un file **FOutj** il cui nome deve risultare dalla concatenazione della stringa "*inverso*" e della stringa corrispondente a **j** (numero d'ordine di creazione del processo figlio). Una volta creato il processo nipote, ogni figlio **Pj** e ogni nipote **PPj** devono leggere, dal loro file associato **Fj+1**, metà del file **un carattere alla volta in senso inverso** e tali caratteri devono essere scritti sul file **FOut** seguendo un precisa **regola**: la regola è che il processo nipote **PPj** deve leggere un carattere alla volta partendo dalla fine del file **Fj+1** fino alla metà del file e dopo ogni lettura deve scrivere il carattere letto sul file **FOutj**; quindi, il processo nipote **PPj** deve comunicare al processo figlio **Pj** che può iniziare la lettura della sua metà e perciò il processo figlio **Pj** deve leggere un carattere alla volta partendo dalla metà - 1 del file fino all'inizio del file e dopo ogni lettura deve scrivere il carattere letto sul file **FOutj**; alla fine, il processo figlio **Pj** deve comunicare al processo padre che la scrittura sul file **FOut** è terminata[1]**.** Il padre ha il compito di stampare su standard output, *rispettando l'ordine dei file* (**F1**, **F2**, ... **FN**), il contenuto dei file **FOutj** avendo cura di stampare anche il nome del file FOutj su una linea separata. Al termine, ogni processo nipote **PPj** deve ritornare al figlio **Pj** il numero di caratteri letti/scritti e la stessa cosa deve fare il processo figlio **Pj** al padre. Il padre, dopo che i figli sono terminati, deve stampare, su standard output, i **PID** di ogni figlio con il corrispondente valore ritornato.

[1] Se si vuole per la sincronizzazione fra nipote-figlio e figlio-padre si possono usare i segnali.