SISTEMI OPERATIVI e
LABORATORIO DI SISTEMI OPERATIVI
(A.A. 13-14) – 19 GENNAIO 2015

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 **ESAME19Gen15_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 **quattro** parametri: il primo e il secondo devono essere nomi assoluti di direttori che identificano due gerarchie (**G1** e **G2**) all'interno del file system, mentre il terzo e il quarto devono essere considerati singoli caratteri (**C1** e **C2**). 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 che abbia nel suo contenuto almeno una istanza del carattere **CX** (**C1** per **G1** e **C2** per **G2**): si riporti il nome assoluto di tali direttori sullo standard output. **Al termine dell'intera esplorazione ricorsiva di G1 e di G2**, si deve verificare che il numero globale di file trovati in **G1** (**F0**, **F1**, ... **FN-1**) sia uguale al numero globale di file trovati in **G2** (**FF0**, **FF1**, ... **FFN-1**): solo in tale caso, si deve invocare la parte in C passando come parametri i nomi assoluti dei file trovati **F0**, **F1**, ... **FN-1, FF0**, **FF1**, ... **FFN-1** e i caratteri **C1 e C2**.

La parte in C accetta un numero variabile pari **2N+2** di parametri maggiore o uguale a 4 (*da controllare che 2***N** *sia pari e sia* ≥ 2) che rappresentano: i primi **2N** nomi assoluti di file **F0**, **F1**, ... **FN-1, FF0, FF1**, ... **FFN-1**, mentre gli ultimi due rappresentano singoli caratteri (**C1** e **C2**) (da controllare). Il processo padre deve generare in **2** cicli successivi **Nprocessi figli** (in totale quindi devono essere generati **2N** processi figli, **P0 … PN-1 e P'0 … P'N-1**): i processi figli **Pi** sono associati ai file **Fi** mentre i processi figli **P'i** sono associati ai file **FFi (con i che, in entrambi i casi, varia da 0 a N-1)**. Ogni figlio **Pi** e **P'i** esegue concorrentemente: in particolare, i primi **N**processi (**Pi**) devono cercare le occorrenze del carattere **C1**, mentre gli altri **N**processi (**P'i**) le occorrenze del carattere **C2**. I processi figli devono attenersi ad uno schema di comunicazione a pipeline ad **N** ad **N** *compreso* il processo padre: in particolare, il figlio **PN-1** comunica con il figlio **PN-2** che comunica con il figlio **PN-3** etc. fino al figlio **P0** che comunica con il padre e la stessa cosa per gli altri **N** figli e cioè il figlio **P'N-1** comunica con il figlio **P'N-2** che comunica con il figlio **P'N-3** etc. fino al figlio **P'0** che comunica con il padre. Le strutture dati che i processi figli **Pi** e **P'i** devono comunicare nelle due pipeline devono avere **3 campi** (il primo deve essere un *int*, mentre gli altri due devono essere *long int*), *pid*, *occmax* e *occtotale*: *pid* deve essere il PID del processo che ha trovato nel suo file associato il massimo numero di occorrenze di **CX** (**C1** o **C2**), *occmax* il valore di tale massimo e *occtotale* il conteggio globale ottenuto fino a quel momento. Il padre ha il compito di stampare su standard output tutti i campi delle due strutture ricevute dalle due pipeline (prima quella dei primi N figli e poi quella degli altri) aggiungendo l'indicazione del carattere **CX** cui si riferiscono con una opportuna spiegazione.