SISTEMI OPERATIVI e
LABORATORIO DI SISTEMI OPERATIVI
(A.A. 14-15) – 12 FEBBRAIO 2016

IMPORTANTE:

LEGGERE LE INFORMAZIONI SUL RETRO DEL FOGLIO!!!

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 **2 parametri**: il primo deve essere il nome assoluto di un direttorio che identifica una gerarchia (**G**) all'interno del file system e il secondo deve essere un carattere alfabetico minuscolo (**Cx**). Il programma deve cercare nella gerarchia **G** specificata tutti i direttori che contengono almeno *un* file leggibile che contenga (nel suo contenuto) almeno una occorrenza del carattere **Cx**. Si riporti il nome assoluto di tali direttori sullo standard output. Al termine dell'intera esplorazione ricorsiva di G, se sono stati trovati almeno **2** file, si deve invocare la parte in C passando come parametri i nomi dei file trovati (**F1, F2, ... FN-1**) -che soddisfano la condizione precedente- e il carattere **Cx**.

La parte in C accetta un numero variabile **N+1** di parametri (con **N** maggiore o uguale a **2**, da controllare**)** che rappresentano **N** nomi di file (**F1, F2. ... FN-1**), mentre l'ultimo rappresenta un carattere alfabetico minuscolo (**Cx**) (da controllare).
Il processo padre deve generare **N processi figli** (**P0, P1, … PN-1**): i processi figli **Pi (con i che varia da 0 a N-1)** sono associati agli **N** file **Fj** (con j= i+1). Ogni processo figlio **Pi** deve leggere i caratteri del file associato **Fj** cercando il carattere **Cx**. I processi figli e il processo padre devono attenersi a questo **schema di comunicazione a pipeline**: il figlio **P0** comunica con il figlio **P1** che comunica con il figlio **P2** etc. fino al figlio **PN-1** che comunica con il **padre**. Questo schema a pipeline deve prevedere l'invio in avanti di un array di **strutture** dati ognuna delle quali deve contenere due campi: 1) *c1*, di tipo int, che deve contenere l'indice d'ordine dei processi; 2) *c2*, di tipo long int, che deve contenere il numero di occorrenze del carattere Cx calcolate dal corrispondente processo. *Gli array di strutture DEVONO essere creati da ogni figlio della dimensione minima necessaria per la comunicazione sia in ricezione che in spedizione.* Quindi la comunicazione deve avvenire in particolare in questo modo: il figlio **P0** passa in avanti (cioè comunica) un array di strutture **A1**, che contiene una sola struttura con *c1* uguale a 0 e con *c2* uguale al numero di occorrenze del carattere **Cx** trovate da **P0** nel file **F1**; il figlio seguente **P1**, dopo aver calcolato numero di occorrenze del carattere **Cx** nel file **F2**, deve leggere (*con una singola read*) l'array **A1** inviato da **P0** e quindi deve confezionare l'array **A2** che corrisponde all'array **A0** aggiungendo all'ultimo posto la struttura con i propri dati e la passa (*con una singola write*) al figlio seguente **P2**, etc. fino al figlio **PN-1**, che si comporta in modo analogo, ma passa al **padre**. Quindi, al processo padre deve arrivare l'array **AN** di **N** strutture (uno per ogni processo **P0 … PN-1)**. Il padre deve riportare i dati di ognuna delle **N** strutture su standard output insieme al **pid** del processo corrispondente e al carattere **Cx**.
Al termine, ogni processo figlio **Pi** deve ritornare al padre il valore intero corrisponde al proprio indice d'ordine (**i**); il padre deve stampare su standard output il PID di ogni figlio e il valore ritornato

IMPORTANTE:

1) Fare il login sui sistemi in modalità Linux usando il proprio **username** e **password**, aprire un browser sulla pagina ftp://lica02.lab.unimo.it/README, copiare il comando presente in un terminale ed eseguirlo rispondendo alle domande proposte: sul Desktop, viene creata automaticamente una directory studente_XXX al cui interno viene creato un file denominato student_data.csv che non va eliminato; infine , dopo avere copiato i propri file da chiavetta, passare in modalità testuale-
2) I file prodotti devono essere collocati nella directory studente_XXX dato che tale directory viene zippata e salvata automaticament sul server ad intervalli di tempo regolari. **ALLA SCADENZA DEL TEMPO A DISPOSIZIONE VERRÀ ATTIVATA UNA PROCEDURA AUTOMATICA DI ESTRAZIONE, PER OGNI STUDENTE DEL TURNO, DEI FILE CONTENUTI NELLA DIRETTORY SPECIFICATA.**
3) Il tempo a disposizione per la prova è di **120 MINUTI** per lo svolgimento di tutto il compito e di **90 MINUTI** per lo svolgimento della sola parte C.
4) Non è ammesso **nessun tipo di scambio di informazioni** né verbale né elettronico, pena la invalidazione della verifica: **all'ingresso deve essere lasciato il/i cellulare/i sulla cattedra e potranno essere ripresi solo all'uscita**.
5) L'assenza di commenti significativi verrà penalizzata, così come la mancanza del makefile!
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.**
7) SI RICORDA CHE IN CASO DI ESITO INSUFFICIENTE è necessario visionare il compito prima di potersi iscrivere a qualunque appello successivo!