SISTEMI OPERATIVI e
LABORATORIO DI SISTEMI OPERATIVI
(A.A. 18-19) – 15 GENNAIO 2020

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 un numero variabile di parametri **W+1** (con **W** maggiore o uguale a 2): il primo parametro deve essere considerato un intero strettamente positivo (**H**), mentre gli altri **W** devono essere **nomi assoluti di directory** che identificano **W** gerarchie (**G1, G2, …**) all'interno del file system. Il comportamento atteso dal programma, dopo il controllo dei parametri, è organizzato in **W** fasi, una per ogni gerarchia.
Il programma, per ognuna delle **W** fasi, deve esplorare la gerarchia **Gg** corrispondente - tramite un file comandi ricorsivo, **FCR.sh** - e deve contare **globalmente per ogni singola** gerarchia **Gg** tutti i file che saranno cercati secondo quanto di seguito specificato. Il file comandi ricorsivo **FCR.sh** deve cercare in ogni gerarchia **Gg** che esista almeno un file (**F)** la cui lunghezza in caratteri sia esattamente uguale a **H**: appena trovato un file che soddisfa la specifica, si deve riportare, contestualmente, il suo nome assoluto sullo standard output. Al termine di OGNUNA delle **W** fasi, si deve riportare sullo standard output il numero di file (**F)** trovati e, solo nel caso tale numero sia *pari* si deve invocare la parte in C, passando come parametri i nomi assoluti dei file ***trovati*** (**F1**, **F2**, ... **FN**).

La parte in C accetta un numero variabile **N** di parametri (con **N** maggiore o uguale a **2** e **pari**, da controllare**)** che rappresentano nomi di file **F1, F2. ... FN** (tutti con uguale lunghezza, che non deve essere controllata). Il processo padre deve generare **N/2** processi figli (P0 ... PN/2-1) e ognuno dei processi figli deve generare un *processo nipote* (PP0 ... PPN/2-1): i processi figli Pi sono associati ai file Fi+1 mentre i processi nipoti PPi ai file FN/2+i+1 (con *i* che, in entrambi i casi, varia da 0 a N/2). Ogni processo figlio Pi deve, *prima di creare il proprio nipote*, creare un file **FOut** il cui nome deve risultare dalla concatenazione della stringa "merge" e della stringa corrispondente a i (numero d'ordine di creazione del processo figlio). Una volta creato il processo nipote, ogni figlio e ogni nipote eseguono concorrentemente; in particolare, ognuno dei due 'tipi' di processi deve leggere, dal suo file associato, un carattere alla volta e quindi lo deve scrivere sul file FOut: **la scrittura deve avvenire in modo strettamente alternato**, iniziando dal figlio Pi. In altre parole, ogni figlio Pi legge il primo carattere dal file Fi+1 e lo scrive sul file FOut e quindi deve comunicare l'avvenuta scrittura sul file al proprio nipote PPi, quindi il processo PPi che ha concorrentemente letto il primo carattere dal file FN/2+i+1 lo può scrivere sul file FOut e può comunicare al proprio figlio l'avvenuta scrittura; tale schema di comunicazione/sincronizzazione[*] deve continuare per tutti i caratteri dei due file associati. Al termine, ogni processo nipote PPi deve ritornare al figlio il valore dell'ultimo carattere scritto nel file FOut e, a sua volta, ogni processo figlio Pi lo deve ritornare 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.

[*] Se si vuole per la sincronizzazione si possono usare i segnali.

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_2_1_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.
   ATTENZIONE: DOPO AVER EVENTUALMENTE USATO LA/LE CHIAVETTA/E, QUESTA/E DEVE/ONO ESSERE PORTATA/E ALLA CATTEDRA E MESSA/E DI FIANCO AL PROPRIO CELLULARE!
2) I file prodotti devono essere collocati nella directory **studente_2_1_XXX** dato che tale directory viene zippata e salvata automaticamentw 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) Per facilitare le operazioni di stampa dei compiti sono imposte le seguenti regole per nominare i file da salvare nella directory **studente_2_1_XXX**:
   - FCP.sh per il file che contiene lo script principale (quello di partenza) della parte SHELL;
   - FCR.sh per il file che contiene lo script ricorsivo della parte SHELL;
   - main.c per il file che contiene il programma della parte C;
   - makefile per il file che contiene le direttive per il comando make.

   Devono essere rispettati esattamente i nomi indicati altrimenti NON si procederà alla correzione del compito!
4) **ATTENZIONE: IN PARTICOLARE PER LA PARTE C NON VERRANNO CORRETTE SOLUZIONI CHE PRESENTANO ERRORI RIPORTATI DAL COMANDO gcc ALL'INVOCAZIONE DEL COMANDO make!!!!**
4) NON devono essere presenti altri file con nome che termina con .sh o con .c nella directory **studente_1_1_USERNAME.**
6) Il tempo a disposizione per la prova è di **120 MINUTI** per il compito completo e di **90 MINUTI** per lo svolgimento della sola parte C.
7) 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**.
8) L'assenza di commenti significativi verrà penalizzata, così come la mancanza del makefile!
9) **AL TERMINE DELLA PROVA È INDISPENSABILE CONSEGNARE IL TESTO DEL COMPITO (ANCHE IN CASO UNO STUDENTE SI RITIRI): IN CASO CONTRARIO, NON POTRÀ ESSERE EFFETTUATA LA CORREZIONE DEL COMPITO MANCANDO IL TESTO DI RIFERIMENTO.**
10) SI RICORDA CHE IN CASO DI ESITO INSUFFICIENTE è necessario visionare il compito prima di potersi iscrivere a qualunque appello successivo!