SISTEMI OPERATIVI E LAB.
(A.A. 22-23) – 6 SETTEMBRE 2023

IMPORTANTE:
SEGUIRE TUTTE LE REGOLE FORNITE PRIMA DELLO SVOLGIMENTO DELL'ESAME!

Esercizio

Si realizzi un programma **concorrente** per UNIX che deve avere una parte in **Bourne Shell** e una parte in **C**.

TESTO PARTE SHELL: ATTENZIONE LEGGERE ANCHE LA NOTA SEGUENTE AL TESTO!

La parte in Shell deve prevedere un numero variabile di parametri **Q+1** (con **Q** maggiore o uguale a **2**): il primo parametro deve essere il **nome assoluto di una directory** che identifica una gerarchia (**G)** all'interno del file system, mentre gli altri **Q** devono essere **nomi relativi semplici di file** (**F1, F2, ...**).
Il comportamento atteso dal programma, dopo il controllo dei parametri, è organizzato in una singola fase per la singola gerarchia **G**.
Il programma deve esplorare la gerarchia **G -** tramite un file comandi ricorsivo, **FCR.sh** – e deve cercare tutte le directory che contengono almeno la **metà (M)*** dei file aventi come nome relativo semplice uno di quelli passato come parametri (**F1, F2, ...**): si riporti il **nome assoluto di tali directory** sullo standard output e quindi si invochi **la parte C** passando come parametri i nomi relativi semplici dei file *trovati* cioè aventi come nome relativo semplice uno di quelli passato come parametri.

NOTA BENE NEI DUE FILE COMANDI SI USI OBBLIGATORIAMENTE:

- una variabile di nome **G** per contenere il primo parametro di FCP.sh;
- una variabile di nome **M** per contenere il numero corrispondente alla metà del numero dei nomi dei file passati come parametri;
- una variabile di nome **F** per identificare, via via, i singoli file delle directory esplorate;
- una variabile di nome **cont** per contare i file trovati via via in una directory;
- una variabile di nome **files** per raccogliere i nomi dei file *trovati* via via in una directory.

OSSERVAZIONE: se per provare la parte shell, si commenta la chiamata alla parte C, ricordarsi di togliere il commento prima della consegna!

TESTO PARTE C: ATTENZIONE LEGGERE ANCHE LA NOTA SEGUENTE AL TESTO!

La parte in C accetta un numero variabile di parametri **N** (con **N** maggiore o uguale a 2) che rappresentano nomi di file (**F1, …FN**).
Il processo padre deve generare **N processi figli**: ogni processo figlio **Pn** è associato ad uno dei file **F1, …FN** (*in ordine*). Ognuno di tali processi figli **Pn** esegue concorrentemente e, per prima cosa, deve calcolare la lunghezza in linee del proprio file associato in termini di *long int*. Quindi, i processi figli **Pn** 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 una singola struttura dati, che deve contenere due campi: 1) *c1*, di tipo *long int*, che deve contenere il valore minimo di linee calcolato dal processo **Pn**; 2) *c2*, di tipo *int*, che deve contenere l'indice d'ordine (**n**) del processo che ha calcolato il minimo (come visto in altri testi di esame) e perciò, al processo padre deve arrivare una singola struttura: il padre deve riportare i dati di tale struttura su standard output insieme al PID del processo che ha calcolato il minimo numero di linee e al nome del file per cui sono stati calcolati (inserendo opportune spiegazioni per l'utente).
Esaurita questa fase a pipeline, ogni processo figlio **Pn** deve aspettare di ricevere il numero minimo di linee (di tipo *long int*) inviato dal padre (secondo quanto specificato in seguito) e, una volta selezionata la linea corrispondente a tale numero, deve stampare su standard output: il suo indice d'ordine, il suo PID, il numero di linea inviato dal padre (che rappresenta il minimo calcolato nella fase precedente), il nome del proprio file associato e quindi la linea selezionata. Il padre, dopo aver ricevuto la struttura, deve comunicare **a tutti** i figli **Pn** il numero minimo di linee ricevuto (che si ricorda è un *long int*): tale valore produrrà una stampa sullo standard output *da parte dei figli*, come spiegato precedentemente.
Al termine dell'esecuzione, ogni figlio **Pn** ritorna al padre il valore intero corrispondente al proprio indice d'ordine (**n**); il padre deve stampare su standard output il PID di ogni figlio e il valore ritornato.

NOTA BENE NEL FILE C main.c SI USI OBBLIGATORIAMENTE:

- una variabile di nome **N** per il numero di processi figli;
- una variabile di nome **n** per l'indice dei processi figli;
- una variabile di nome **linea per l'array di 250 caratteri** per le linee lette dai figli dal proprio file; *si supponga che 250 caratteri siano sufficienti per ogni linea, compreso il terminatore di linea/stringa.*
- una variabile di nome **cur_tot** per il calcolo del numero di linee effettuato dai figli;
- una variabile di nome **min** per il valore minimo ricevuto dai figli da parte del padre.

* Per metà M si intende il risultato della divisione intera di Q per 2: quindi se Q=5 allora M=2!