

SISTEMI OPERATIVI/SISTEMI OPERATIVI E LAB.

(A.A. 23-24) – 15 GENNAIO 2025

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+2** (con **Q** maggiore o uguale a 2): i primi due parametri devono essere considerati numeri interi strettamente positivi (**N** e **X**, con **N** strettamente minore di X), mentre gli altri **Q** devono essere **nomi assoluti di directory** che identificano **Q** gerarchie (**G1, G2, ...**) all'interno del file system. Il comportamento atteso dal programma, dopo il controllo dei parametri, è organizzato in **Q** fasi, una per ogni gerarchia. Il programma, per ognuna delle **Q** fasi, deve esplorare la gerarchia **G** corrispondente - tramite un file comandi ricorsivo, **FCR.sh** - e deve cercare tutte le directory che contengono *almeno* un file leggibile con lunghezza in linee compresa fra **N** e **X** (*estremi inclusi*): si riporti il nome assoluto di tali directory sullo standard output. In ognuna di tali directory, per ogni file trovato che soddisfa la condizione precedente, si deve invocare la parte in C, passando come parametri il nome del file trovato *corrente* e il numero **N**.

NOTA BENE NEI DUE FILE COMANDI SI USI OBBLIGATORIAMENTE:

- una variabile di nome **N** per contenere il primo parametro di FCP.sh;
- una variabile di nome **X** per contenere il secondo parametro di FCP.sh;
- una variabile di nome **G** per le singole gerarchie di ognuna delle **Q** fasi;
- una variabile di nome **F** per identificare, via via, i singoli file delle directory esplorate.

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

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

La parte in C accetta esattamente due parametri: il primo rappresenta il nome di un file (**F**), mentre il secondo deve essere considerato un numero intero strettamente positivo (**N**, da controllare). Il processo padre deve, *per prima cosa*, creare nella directory di sistema **/tmp** un file con nome **RISULTATO**, se non esiste, altrimenti deve aprire tale file in append. Quindi, il processo padre deve generare un numero di **processi figli** pari a **N**: ogni processo figlio **Pn** è associato ad una delle linee del file **F** (*in ordine*); quindi, il figlio **P0** è associato alla prima linea (*di numero 1*), il figlio **P1** alla seconda linea (*di numero 2*) e così via.

Ognuno di tali processi figli **Pn** esegue concorrentemente e legge tutte le linee del proprio file associato per selezionare la linea di propria competenza: selezionata tale linea, ogni processo **Pn** deve scrivere tale linea (insieme con il terminatore di linea) sul file **/tmp/RISULTATO**. I processi figli devono sincronizzarsi a vicenda in modo che le scritture sul file **/tmp/RISULTATO** avvengano **in ordine** dal primo processo all'ultimo fino a che si selezionano linee dal file **F**. Si usi per i processi figli **Pn** uno schema di sincronizzazione a pipeline, che però risulta un po' degenerare per due ragioni: a) nella pipeline non c'è il padre; b) l'ultimo figlio non deve mandare a nessuno: il generico processo **Pn** (a parte il processo **P0**) dopo aver ricevuto l'ok dal figlio precedente stampa quanto richiesto e poi manda l'ok al figlio successivo (a parte il processo **PN-1**).

Al termine dell'esecuzione, ogni figlio **Pn** ritorna al padre il numero totale di linee del file **F** (*supposto minore di 255*); il padre deve stampare su standard output il PID di ogni figlio e il valore ritornato.

OSSERVAZIONE: questa parte C realizza una versione parallela del comando **head** con **N** come numero di linee da visualizzare a partire dall'inizio del file e con scrittura delle linee su file **/tmp/RISULTATO** invece che su standard output!

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 **outfile** per il file descriptor usato dal padre per la creazione/apertura del file **/tmp/RISULTATO**;
- una variabile di nome **linea** per l'array di 250 caratteri usato per leggere, da parte dei figli, via via le singole linee; si supponga che 250 caratteri siano sufficienti per ogni linea, compreso il terminatore di linea.