SISTEMI OPERATIVI/SISTEMI OPERATIVI E LAB.
(A.A. 24-25) – 11 GIUGNO 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 **N+1** (con **N maggiore o uguale a 2**): il primo parametro deve essere il **nome assoluto di una directory** che identifica una gerarchia all'interno del file system (**G**), mentre gli altri **N** devono essere considerati singoli caratteri (**C1, …CN**). Il comportamento atteso dal programma, dopo il controllo dei parametri, è organizzato in una singola fase.
Il programma deve esplorare la gerarchia **G** - tramite un file comandi ricorsivo, **FCR.sh** - e deve contare globalmente tutti i file che rispettano la seguente specifica: i file devono essere **leggibili** e devono contenere (nel contenuto) almeno una occorrenza di **TUTTI** i caratteri **C1, …CN**; **per ogni directory analizzata** si deve riportare il suo nome assoluto, insieme con l'indicazione se è stato trovato almeno un file che soddisfa la precedente specifica oppure no.
Al termine dell'unica fase, si deve riportare sullo standard output il numero totale di file trovati (che soddisfano la condizione sopra indicata). Quindi, **per ognuno dei file trovati F**, si deve invocare la parte in C, passando come parametri **F** e i caratteri **C1, …CN**.

NOTA BENE NEI DUE FILE COMANDI SI USI OBBLIGATORIAMENTE:

- una variabile di nome **G** per il primo parametro di FCP.sh;
- il nome **/tmp/nomiAssoluti** per il nome del file temporaneo.
- una variabile di nome **F** per identificare, via via, i singoli file delle directory esplorate e quelli per i quali si deve invocare la parte C;
- una variabile di nome **count** per contare, localmente a FCR.sh, i file che soddisfano la specifica;

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+1** (con **N maggiore o uguale a 2**), **F, C1, …CN**, che rappresentano rispettivamente le seguenti informazioni: il primo il nome assoluto di un file (**F**) e gli ultimi **N** devono essere considerati singoli caratteri (*da controllare*). Il processo padre deve generare un numero di **processi figli** pari a **N**: ogni processo figlio **Pn** è associato ad uno dei caratteri **C1, …CN** (*in ordine*).
Ognuno di tali processi figli **Pn** esegue concorrentemente e calcola il numero di occorrenze (come *long int*) del proprio carattere associato presenti nel file **F**. 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 un array **cur** di **strutture** dati di tipo **Strut**; ogni struttura deve contenere due campi: 1) *c1*, di tipo *int*, che deve contenere il pid del processo figlio **Pn**; 2) *c2*, di tipo *long int*, che deve contenere il numero di occorrenze calcolate dal corrispondente processo figlio. *Gli array di strutture DEVONO essere creati da ogni figlio della dimensione minima necessaria per la comunicazione sia in ricezione che in spedizione* (come visto in altri testi di esame). Quindi, al processo padre deve arrivare l'array **cur** di strutture (una per ogni processo **P0** … **PN-1)**. Il padre deve riportare i dati di ognuna delle **N** strutture su standard output insieme al numero d'ordine **n** del processo figlio corrispondente, al carattere associato e al nome del file **F**.
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;
- un tipo di nome **Strut** per le strutture dati;
- una variabile di nome **cur** (i cui elementi devono essere di tipo **Strut)** per l'array usato sia dai figli che dal padre.