SISTEMI OPERATIVI E LAB.
(A.A. 22-23) – 7 GIUGNO 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 considerato un numero **intero** (**X**) **strettamente positivo e strettamente minore di 4**, 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 si trovano a livello corrispondente al numero **X**. Si riporti il nome assoluto di tali directory sullo standard output. In ogni directory trovata, si deve invocare la parte in C passando come parametri i nomi di tutti i file leggibili che *NON* siano vuoti (**F1, F2, ...**) presenti in tale direttorio.

NOTA BENE NEI DUE FILE COMANDI SI USI OBBLIGATORIAMENTE:
- una variabile di nome **X** per contenere il primo 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 trovate a livello **X**.

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** maggiore o uguale a **2** che rappresentano nomi di file (**F1, …FN**): si assuma (senza bisogno di effettuare alcun controllo) che i file NON siano vuoti.
Il processo padre deve generare un numero di **processi figli** pari a **N**: ogni processo figlio **Pn** è associato ad uno dei file **F1, …FN** (*in ordine*). Ognuno di tali processi figli **Pn** esegue concorrentemente e legge tutte le linee del proprio file associato operando una opportuna selezione come indicato nel seguito. Dopo la lettura di ogni linea, il processo figlio **Pn** deve comunicare al padre l'ultimo carattere della linea corrente (prima del terminatore di linea) e deve ricevere dal padre l'indicazione di stampare o meno su standard output delle informazioni (*vedi dopo* *). Il padre deve ricevere, rispettando l'ordine dei file **F1, …FN**, da ogni figlio via via i caratteri che rappresentano l'**ultimo** carattere della linea corrente. Quindi, al processo padre deve arrivare **un insieme di caratteri** (che via via potrebbe diminuire in quantità in dipendenza della terminazione dei figli): sicuramente il primo insieme (questo viene garantito dalla parte Shell) è costituito dall'ultimo carattere della prima linea inviato dal figlio **P0**, dall'ultimo carattere della prima linea inviato dal figlio **P1**, … , dall'ultimo carattere della prima linea inviato dal figlio **PN-1**. Per ogni insieme ricevuto, il padre deve determinare il valore **massimo** e, SOLO AL PROCESSO FIGLIO CHE HA INVIATO TALE VALORE, deve indicare *(*)* di stampare su standard output **l'indice d'ordine del processo, il suo PID, il carattere identificato come massimo e quindi la linea corrente**, mentre a tutti gli altri processi figli deve indicare di non stampare♦.
Al termine, ogni processo figlio **Pn** deve ritornare al padre il numero di linee che ha scritto sullo standard output (supposto minore di 255) e il padre deve stampare su standard output i 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 le linee che via via devono leggere i processi figli;
- una variabile di nome **car** per i caratteri che via via deve leggere il processo padre.

OSSERVAZIONI DERIVANTI DA RISPOSTE DATE DURANTE L'ESAME:
1) Per una dimenticanza NON è stata inserita l'ipotesi della lunghezza massima della linea e quindi bisognava inserire un commento significativo in corrispondenza della definizione della variabile **linea** come array statico di char.
2) Il fatto che i file non siano vuoti non escludeva il caso di avere linee vuote e cioè costituite solo dallo '\n': è stato chiarito che si poteva ipotizzare che questo NON avvenisse mai e che quindi tutte le linee di tutti i file contenessero almeno un altro carattere oltre lo '\n'!

♦ Per questo tipo di interazione, volendo, si possono usare i segnali. Nel caso si usino le pipe, fare attenzione che il padre deve inviare l'indicazione SOLO ai figli che non sono terminati!