SISTEMI OPERATIVI E LAB.
(A.A. 19-20) – 9 SETTEMBRE 2020

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 assoluti di una directory** che identifica una gerarchia all'interno del file system (**G**), mentre gli altri **Q** devono essere considerati singoli caratteri (**C1, …CQ**). 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 cercare tutti i file che contengono (nel contenuto) almeno una occorrenza di TUTTI i caratteri **C1, …CQ**. Si riporti il **nome assoluto** di ognuno di tali file **F** sullo standard output e, contestualmente, si deve invocare la parte in C, passando come parametri **F, L** e i caratteri **C1, …CQ**, dove **L** è la lunghezza in linee del file **F**.

NOTA BENE NEI DUE FILE COMANDI SI USI OBBLIGATORIAMENTE:

- una variabile di nome **G** per il primo parametro;
- una variabile di nome **F** per identificare, via via, i singoli file per i quali si deve invocare la parte C;
- una variabile di nome **L** per la lunghezza in linee del file corrente.

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 **Q+2** (con **Q maggiore o uguale a 2**), **F, L** e **C1, …CQ**: rappresentano rispettivamente le seguenti informazioni: un nome di file e un numero strettamente positivo (da controllare) che rappresenta il numero di linee del file e gli ultimi **Q** devono essere considerati singoli caratteri. Il processo padre deve generare un numero di **processi figli** pari a **Q**: ogni processo figlio **Pq** è associato ad uno dei caratteri **C1, …CQ** (*in ordine*).
Ognuno di tali processi figli **Pq** esegue concorrentemente e calcola il numero di occorrenze del proprio carattere associato, per ognuna delle **L** linee del file **F**. I processi padre e figli devono ***sincronizzarsi strettamente***[*] in modo che, per ognuna delle **L** linee del file **F**, sullo standard output siano scritte le seguenti informazioni:
- il padre deve riportare il numero di linea correntemente analizzata da tutti i processi figli (Nota bene: la numerazione delle linee deve essere fatta partire da 1), con opportune frasi;
- il primo figlio (**P0**) deve riportare il numero di occorrenze del proprio carattere (**C1**) trovate nella linea corrente, con opportune frasi;
- il secondo figlio (**P1**) deve riportare il numero di occorrenze del proprio carattere (**C1**) trovate nella linea corrente, con opportune frasi;
- *così via per tutti i figli*;
- *quindi così via per tutte le **L** linee del file **F**.*

ESEMPIO DI OUTPUT, SUPPONENDO DI AVERE 2 CARATTERI PASSATI E 2 LINEE:

```
Linea 1:
1 occorrenze del carattere 'S'
2 occorrenze del carattere 'n'
Linea 2:
2 occorrenze del carattere 'S'
7 occorrenze del carattere 'n'
```

Al termine dell'esecuzione, ogni figlio **Pq** ritorna al padre il numero (*supposto strettamente minore di 255*) di occorrenze trovate nell'ultima linea del file **F**; 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 **Q** per il numero di processi figli;
- una variabile di nome **q** per l'indice dei processi figli;
- una variabile di nome **L** per la lunghezza in linee del file **F**.

[*] Volendo per questo tipo di interazione si possono usare i segnali.