

# SISTEMI OPERATIVI e LABORATORIO DI SISTEMI OPERATIVI (A.A. 18-19) – 19 GIUGNO 2019

**IMPORTANTE:** LEGGERE LE INFORMAZIONI SUL RETRO DEL FOGLIO!!!

## Esercizio

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

La parte in Shell deve prevedere un numero variabile di parametri **W+1** (con **W** maggiore o uguale a 2): il primo parametro **H** deve essere considerato un numero intero strettamente maggiore di 1 e minore di **255**, mentre gli altri **W** devono essere **nomi assoluti di directory** che identificano **W** gerarchie (**G1, G2, ...**) all'interno del file system. Il comportamento atteso dal programma, dopo il controllo dei parametri, è organizzato in **W** fasi, una per ogni gerarchia.

Il programma, per ognuna delle **W** fasi, deve esplorare la gerarchia **Gg** corrispondente - tramite un file comandi ricorsivo, **FCR.sh** - e deve cercare tutti i direttori che contengono almeno due file che abbiano un numero di linee strettamente minore di **H**: si riporti il nome assoluto di tali direttori sullo standard output. In ogni direttorio trovato, si deve invocare la parte in C, passando come parametri i **nomi** dei file trovati (**F1, F2, ... FN**).

La parte in C accetta un numero variabile **N** di parametri (con **N** maggiore o uguale a **2**, da controllare) che rappresentano **N** nomi di file (**F1, F2, ... FN**).

Il processo padre deve generare **N** processi figli (**P0, P1, ... PN-1**): i processi figli **Pi** (con **i** che varia da **0** a **N-1**) sono associati agli **N** file **Ff** (con **f = i+1**). Ogni processo figlio **Pi** deve leggere tutte le linee<sup>o</sup> del file associato ~~**Ff cercando le occorrenze del carattere**~~; dopo la lettura di ogni linea, il processo figlio **Pi** deve comunicare al padre il primo carattere della linea corrente 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 **Ff**, da ogni figlio via via i caratteri che rappresentano il primo 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 dal primo carattere della prima linea inviato dal figlio **P0**, dal primo carattere della prima linea inviato dal figlio **P1**, ..., dal primo 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<sup>♦</sup>. Al termine, ogni processo figlio **Pi** deve ritornare al padre il numero di linee che hanno scritto sullo standard output e il padre deve stampare su standard output i PID di ogni figlio e il valore ritornato.

---

<sup>o</sup> Ogni linea si può supporre che abbia una lunghezza massima di 250 caratteri, compreso il terminatore di linea e, se serve, il terminatore di stringa.

<sup>♦</sup> 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!

# SISTEMI OPERATIVI e LABORATORIO DI SISTEMI OPERATIVI (A.A. 18-19) – 19 GIUGNO 2019

**IMPORTANTE:** LEGGERE LE INFORMAZIONI SUL RETRO DEL FOGLIO!!!

## Esercizio

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

La parte in Shell deve prevedere un numero variabile di parametri **Z+1** (con **Z** strettamente maggiore di 1): il primo parametro **K** deve essere considerato un numero intero strettamente maggiore di 1 e minore di **255**, mentre gli altri **Z** devono essere **nomi assoluti di directory** che identificano **Z** gerarchie (**G1**, **G2**, ...) all'interno del file system. Il comportamento atteso dal programma, dopo il controllo dei parametri, è organizzato in **Z** fasi, una per ogni gerarchia.

Il programma, per ognuna delle **Z** fasi, deve esplorare la gerarchia **Gg** corrispondente - tramite un file comandi ricorsivo, **FCR.sh** - e deve cercare tutti i direttori che contengono almeno due file che abbiano un numero di linee strettamente minore di **K**: si riporti il nome assoluto di tali direttori sullo standard output. In ogni direttorio trovato, si deve invocare la parte in C, passando come parametri i **nomi** dei file trovati (**F1**, **F2**, ... **FM**).

La parte in C accetta un numero variabile **M** di parametri (con **M** maggiore o uguale a **2**, da controllare) che rappresentano **M** nomi di file (**F1**, **F2**, ... **FM**).

Il processo padre deve generare **M processi figli** (**P0**, **P1**, ... **PM-1**): i processi figli **Pj** (con **j** che varia da **0** a **M-1**) sono associati agli **M** file **Ff** (con **f = j+1**). Ogni processo figlio **Pj** deve leggere tutte le linee<sup>o</sup> del file associato **Ff** ~~creando le occorrenze del carattere~~; dopo la lettura di ogni linea, il processo figlio **Pj** deve comunicare al padre il primo carattere della linea corrente 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 **Ff**, da ogni figlio via via i caratteri che rappresentano il primo 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 dal primo carattere della prima linea inviato dal figlio **P0**, dal primo carattere della prima linea inviata dal figlio **P1**, ... , dal primo carattere della prima linea inviata dal figlio **PM-1**. Per ogni insieme ricevuto, il padre deve determinare il valore **minimo** 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 minimo e quindi la linea corrente**, mentre a tutti gli altri processi figli deve indicare di non stampare<sup>♦</sup>. Al termine, ogni processo figlio **Pj** deve ritornare al padre il numero di linee che hanno scritto sullo standard output e il padre deve stampare su standard output i PID di ogni figlio e il valore ritornato.

**IMPORTANTE:**

---

<sup>o</sup> Ogni linea si può supporre che abbia una lunghezza massima di 250 caratteri, compreso il terminatore di linea e, se serve, il terminatore di stringa.

<sup>♦</sup> 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!

- 1) Fare il login sui sistemi in modalità Linux usando il proprio **username** e **password**, aprire un browser sulla pagina <ftp://lica02.lab.unimo.it/README>, copiare il comando presente in un terminale ed eseguirlo rispondendo alle domande proposte: sul Desktop, viene creata automaticamente una directory **studente\_1\_1\_XXX** al cui interno viene creato un file denominato **student\_data.csv** che non va eliminato; infine, dopo avere copiato i propri file da chiavetta, passare in modalità testuale.
- 2) I file prodotti devono essere collocati nella directory **studente\_1\_1\_XXX** dato che tale directory viene zippata e salvata automaticamente sul server ad intervalli di tempo regolari. **ALLA SCADENZA DEL TEMPO A DISPOSIZIONE VERRÀ ATTIVATA UNA PROCEDURA AUTOMATICA DI ESTRAZIONE, PER OGNI STUDENTE DEL TURNO, DEI FILE CONTENUTI NELLA DIRETTORY SPECIFICATA.**
- 3) **NOVITÀ DALL'APPELLO DI LUGLIO 2016:** per facilitare le operazioni di stampa dei compiti sono imposte le seguenti regole per nominare i file da salvare nella directory **studente\_1\_1\_USERNAME**:
  - FCP.sh per il file che contiene lo script principale (quello di partenza) della parte SHELL;
  - FCR.sh per il file che contiene lo script ricorsivo della parte SHELL;
  - main.c per il file che contiene il programma della parte C;
  - makefile per il file che contiene le direttive per il comando make.

**Devono essere rispettati esattamente i nomi indicati altrimenti NON si procederà alla correzione del compito!**
- 4) NON devono essere presenti altri file con nome che termina con .sh o con .c nella directory **studente\_1\_1\_USERNAME**.
- 5) Il tempo a disposizione per la prova è di **120 MINUTI** per il compito completo e di **90 MINUTI** per lo svolgimento della sola parte C.
- 6) Non è ammesso **nessun tipo di scambio di informazioni** né verbale né elettronico, pena la invalidazione della verifica: **all'ingresso deve essere lasciato il/i cellulare/i sulla cattedra e potranno essere ripresi solo all'uscita.**
- 7) L'assenza di commenti significativi verrà penalizzata, così come la mancanza del makefile!
- 8) **AL TERMINE DELLA PROVA È INDISPENSABILE CONSEGNARE IL TESTO DEL COMPITO (ANCHE IN CASO UNO STUDENTE SI RITIRI): IN CASO CONTRARIO, NON POTRÀ ESSERE EFFETTUATA LA CORREZIONE DEL COMPITO MANCANDO IL TESTO DI RIFERIMENTO.**
- 9) **SI RICORDA CHE IN CASO DI ESITO INSUFFICIENTE è necessario visionare il compito prima di potersi iscrivere a qualunque appello successivo!**

