SISTEMI OPERATIVI e
LABORATORIO DI SISTEMI OPERATIVI
(A.A. 15-16) – 8 GIUGNO 2016

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 **4 parametri**: i primi due devono essere nomi assoluti di due direttori che identificano due gerarchie (**G1** e **G2**) all'interno del file system, il terzo deve essere il nome relativo semplice di un direttorio (**D**), mentre il quarto parametro deve essere considerato un numero intero strettamente positivo e minore di 255 (**H**). Il programma deve cercare in due fasi successive, prima nella gerarchia **G1** e poi nella gerarchia **G2** specificate, tutti i direttori di nome **D** che contengono almeno *due* file che abbiano un numero di linee uguale a **H**. Si riporti il nome assoluto di tali direttori sullo standard output. Al termine dell'intera esplorazione ricorsiva di **G1** e **G2**, solo se è stato trovato almeno* un direttorio di nome **D** in ognuna delle due gerarchie (quindi almeno due file in ognuna delle due gerarchie) si deve invocare la parte in C, passando come parametri i nomi assoluti dei file trovati (**F0, F1, ... FN-1**) nei direttori di nome **D** (come sopra indicato) e il numero intero **H**.

La parte in C accetta un numero variabile **N+1** di parametri (con **N** maggiore o uguale a 4**)** che rappresentano i primi **N** nomi di file (**F0, F1, ... FN-1**), mentre l'ultimo rappresenta un numero intero (**H**) strettamente positivo e minore di 255 (da controllare) che indica la lunghezza in linee dei file: infatti, la lunghezza in linee dei file è la stessa (questo viene garantito dalla parte shell e NON deve essere controllato).
Il processo padre deve, per prima cosa, inizializzare il seme per la generazione random di numeri (come illustrato nel retro del foglio) e deve creare un file di nome "/tmp/creato" (**Fcreato**). Il processo padre deve, quindi, generare **N processi figli** (**P0** … **PN-1**): i processi figli **Pi (con i che varia da 0 a N-1)** sono associati agli **N** file **Fk** (con k= **i**+1). Ogni processo figlio **Pi** deve leggere le linee del file associato **Fk sempre** fino alla fine. I processi figli **Pi** e il processo padre devono attenersi a questo **schema di comunicazione**: per ogni linea letta, il figlio **Pi** deve comunicare al padre la lunghezza della linea corrente compreso il terminatore di linea (come int); il padre usando in modo opportuno la funzione mia_random() (riportata nel retro del foglio) deve individuare in modo random°, appunto, quale lunghezza (come int) considerare fra quelle ricevute, rispettando l'ordine dei file, da tutti i figli **Pi**; individuata questa lunghezza, usando sempre in modo opportuno la funzione mia_random() deve individuare un intero che rappresenterà un indice per la linea della lunghezza considerata ; il padre deve comunicare indietro a tutti i figli **Pi** tale indice: ogni figlio **Pi** ricevuto l'indice (per ogni linea) deve controllare se è ammissibile per la linea corrente e in tal caso deve scrivere il carattere della linea corrente, corrispondente a tale indice, sul file **Fcreato**, altrimenti non deve fare nulla e deve passare a leggere la linea successiva.
Al termine, ogni processo figlio **Pi** deve ritornare al padre il valore intero corrispondente al numero di caratteri scritti sul file **Fcreato** (sicuramente minore di 255); il padre deve stampare su standard output il PID di ogni figlio e il valore ritornato.

---

* Precisazione fatta durante lo svolgimento della prova!

° Precisazione fatta durante lo svolgimento della prova: la funzione mia_random() deve essere chiamata due volte dal padre; la prima volta con il numero di processi (in questo caso N); la prima chiamata serve per individuare il figlio di cui deve essere considerata la lunghezza inviata; la seconda volta con la lunghezza individuata al passo precedente!

IMPORTANTE:

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_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_XXX dato che tale directory viene zippata e salvata automaticament 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) Il tempo a disposizione per la prova è di **120 MINUTI** per lo svolgimento di tutto il compito e di **90 MINUTI** per lo svolgimento della sola parte C.
4) Non è ammesso **nessun tipo di scambio di informazioni** né verbale né elettronico, pena la invalidazione della verifica.
5) L'assenza di commenti significativi verrà penalizzata, così come la mancanza del makefile!
6) **AL TERMINE DELLA PROVA È INDISPENSABILE CONSEGNARE IL TESTO DEL COMPITO (ANCHE IN CASO CHE UNO STUDENTE SI RITIRI): IN CASO CONTRARIO, NON POTRÀ ESSERE EFFETTUATA LA CORREZIONE DEL COMPITO MANCANDO IL TESTO DI RIFERIMENTO.**

Chiamata alla funzione di libreria per inizializzare il seme:

```
#include <time.h>

srand(time(NULL));
```

Funzione che calcola un numero random compreso fra 0 e n-1:

```
#include <stdlib.h>

int mia_random(int n)
{
int casuale;
casuale = rand() % n;
return casuale;
}
```