SISTEMI OPERATIVI e LAB.
(A.A. 12-13) – 19 GIUGNO 2013

IMPORTANTE:

1) Fare il login sui sistemi in modalità Linux usando il proprio **username** e **password**, attivare syncexam.sh e passare in modalità testuale.
2) I file prodotti devono essere collocati in un **sottodirettorio** (che deve essere nella directory studente_XXX) che deve essere creato e avere nome **ESAME19Giu13_1_1.** FARE ATTENZIONE AL NOME DEL DIRETTORIO, in particolare alle maiuscole e ai trattini indicati. Verrà penalizzata l'assenza del direttorio con il nome indicato e/o l'assenza dei file nel direttorio specificato, al momento della copia automatica del direttorio e dei file. **ALLA SCADENZA DEL TEMPO A DISPOSIZIONE VERRÀ INFATTI ATTIVATA UNA PROCEDURA AUTOMATICA DI COPIA, PER OGNI STUDENTE DEL TURNO, DEI FILE CONTENUTI NEL DIRETTORIO SPECIFICATO.**
3) Il tempo a disposizione per la prova è di **75 MINUTI** per lo svolgimento della sola parte C e di **120 MINUTI** per lo svolgimento di tutto il compito.
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.
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.**

Esercizio

Si realizzi un programma **concorrente** per UNIX che deve avere una parte in **Bourne Shell** (già realizzata per chi sostiene la sola parte C e da realizzare per chi sostiene la prova completa) e una parte in **C**.

La parte in Shell deve prevedere tre parametri: il primo deve essere il **nome assoluto di un direttorio** che identifica una gerarchia (**G**) all'interno del file system; il secondo e il terzo parametro devono essere considerati entrambi numeri interi **K** e **Y**, con **K** maggiore o uguale a 2 e **Y** strettamente positivo. Il programma deve cercare nella gerarchia **G** specificata tutti i direttori che contengono almeno **K** file con lunghezza in byte pari a **Y** e contenenti solo caratteri numerici. 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**.

La parte in C accetta un numero variabile di parametri che rappresentano **N** nomi di file **F1..FN**: tutti i file **Fi** hanno uguale lunghezza in byte e contengono solo caratteri numerici (queste due proprietà sono garantite dalla parte shell e non devono essere controllate).
Il processo padre deve generare **N processi figli** (**P0** … **PN-1**): ogni processo figlio è associato ad uno dei file **Fi**. Ognuno di tali processi figli esegue concorrentemente, legge tutti i caratteri del file associato **Fi** e calcola per ogni carattere numerico letto il numero intero positivo corrispondente. I processi figli e il processo padre devono attenersi ad uno **schema di comunicazione a pipeline**; il figlio **PN-1** comunica con il figlio **PN-2** etc. fino al figlio **P0** che comunica con il **padre**; questo schema a pipeline deve essere ripetuto per ogni carattere di ogni file e deve prevedere l'invio indietro, per ogni carattere, via via di una struttura **Si** che deve contenere due campi, *c1* e *c2*, con *c1* uguale al PID di un figlio e con *c2* uguale ad un numero intero positivo. In particolare, l'ultimo processo **PN-1** *per ogni carattere numerico letto* dopo aver calcolato il numero corrispondente (*NumeroN-1*), passa indietro (cioè comunica) una struttura **SN-1** con il proprio *PID* e *NumeroN-1*; il processo precedente **PN-2**, dopo aver calcolato il numero (*NumeroN-2*) corrispondente al carattere numerico corrente, riceve da **PN-1** l'informazione comunicata e passa al processo **PN-3** l'informazione ricevuta da **PN-1** se *NumeroN-1* è maggiore o uguale di *NumeroN-2* oppure la propria informazione e così via fino a che il primo processo **P0**, dopo aver calcolato il numero (*Numero0*) corrispondente al carattere numerico corrente, riceve da **P1** l'informazione comunicata e passa al processo **padre** l'informazione ricevuta da **P1** se *Numero1* è maggiore o uguale di *Numero0* oppure la propria informazione. Quindi, al processo padre devono arrivare tante strutture quanti sono i caratteri numerici dei file letti dai processi **P0, P1, … PN-1**. Il padre ha il compito di stampare su standard output tutte le informazioni ricevute dal processo **P0**, che rappresentano il numero corrispondente con massimo valore e il PID del processo che lo ha calcolato per ognuno dei caratteri numerici presenti nei file letti.
Al termine, ogni processo figlio **Pi** deve ritornare al padre l'ultimo carattere numerico letto dal proprio file associato **Fi** e il padre deve stampare su standard output i PID e i valori ritornati dai figli.