

Reti di Calcolatori

Appello del 02/09/2024

Compito 1

Cognome:
Nome:
Matricola:

Tempo a disposizione: 3h

È obbligatorio inserire Cognome Nome Matricola e Numero Compito all'inizio di ogni file sorgente, pena la non valutazione del compito, che viene stampato in modo automatico solo in caso siano presenti gli elementi detti sopra.

Si devono consegnare **tutti i file sorgente e tutti gli eseguibili prodotti singolarmente** (per favore, solo quelli relativi ai file sorgente consegnati!!!).

La prova intende valutare le capacità progettuali e di programmazione sia in **ambiente Java** che in **ambiente C**, pertanto è consigliato sviluppare **entrambe** le soluzioni richieste al meglio. **In entrambi gli esercizi, sia in Java che in C, si effettuino gli opportuni controlli sui parametri della richiesta e si gestiscano le eccezioni, tenendo presente i criteri secondo cui si possa ripristinare il funzionamento del programma oppure si debba forzarne la terminazione. Leggete con attenzione le specifiche del problema prima di impegnarvi "a testa bassa" nello sviluppo delle singole parti. Naturalmente, ci aspettiamo che i componenti da consegnare siano stati provati e siano funzionanti.**

Si richiede la progettazione e la realizzazione di **servizi invocabili su file di testo sul file system di una macchina server** (direttorio remoto), **da parte di utenti da macchine client**. Ogni comando considera come direttorio di partenza il direttorio corrente dove sono stati lanciati i processi interessati. Se, per esempio, il client richiede il numero di righe di un file nel direttorio corrente, il server cerca il file indicato nel direttorio da cui è stato lanciato. Sono oggetti dei servizi i **solli file di testo**, definiti come file con suffisso ".txt" e che siano composti di linee di lunghezza massima di 256 caratteri ASCII stampabili, terminate sempre da fine linea.

In particolare si vogliono realizzare i seguenti servizi:

1. **Conteggio delle linee che iniziano con una lettera maiuscola e contengono almeno uno specificato numero di occorrenze di un carattere indicato dall'utente all'interno di tutti i file di testo presenti sul direttorio remoto:** questa operazione richiede all'utente *un carattere maiuscolo* e il *numero di occorrenze*, quindi scandisce ciascun file di testo riga per riga contando il numero di righe che iniziano con una lettera maiuscola e contengono (in qualsiasi posizione della riga) un numero di occorrenze del carattere uguale o superiore a quelle indicate dall'utente; infine, restituisce a console l'esito dell'operazione.
2. **Eliminazione di tutte le occorrenze di caratteri numerici all'interno di un file di testo:** questa operazione richiede all'utente il *nome di un file*, elimina tutte le occorrenze di *caratteri numerici* (da '0' a '9') dal file stesso, e visualizza sul cliente a video il numero di eliminazioni effettuate dal server o un'indicazione di errore.
3. **Lista dei file di testo di un direttorio specificato dall'utente:** questa operazione richiede all'utente un *nome di direttorio*, quindi visualizza la lista dei *file di testo* (con suffisso ".txt") contenuti nel direttorio richiesto.
4. **Trasferimento dal server al client di tutti i file di un direttorio che iniziano con un specifico prefisso** (al max 4 caratteri alfabetici 'a'-'z' -'A'-'Z'): questa operazione richiede all'utente il *nome del direttorio* e il *prefisso*, e trasferisce i file (**get** di quel direttorio dal server) il cui nome comincia con il prefisso dato, inviandoli come nome e contenuto sul direttorio del client.

Si richiede inoltre di **non** usare comandi Unix (es. il comando **ls**), ma solo primitive di sistema (es. **opendir()** in C) o funzioni di libreria (es. metodi della **classe File** in Java). Inoltre, si considerino e si segnalino le possibilità di **interferenze** fra le operazioni, evitandole dove necessario

Parte Java

Utilizzando **java RMI** sviluppare un'applicazione C/S che consenta di effettuare le operazioni remote per:

- **visualizzazione della lista dei file di un direttorio;**
- **eliminazione di tutte le occorrenze di linee** che contengono una specifica parola all'interno di un file di testo.

Il progetto RMI si basa su un'interfaccia (contenuta nel file *RMI_interfaceFile.java*) in cui vengono definiti i metodi invocabili in remoto dal client:

- Il metodo **lista_filetesto** accetta come parametro d'ingresso il **nome del direttorio**, e restituisce i **nomi dei file di testo trovati, se ci sono**. In caso di direttorio inesistente, prevedere una segnalazione di errore.
- Il metodo **elimina_linee_contenenti_parola** accetta come parametro in ingresso il *nome del file e la parola* e ne restituisce il *numero di linee eliminate*, oppure -1 in caso il file non esista o non contenga la parola cercata.

Si progettino le classi:

- **RMI_Server** (contenuta nel file *RMI_Server.java*), che implementa i metodi del server invocabili in remoto;
- **RMI_Client** (contenuta nel file *RMI_Client.java*), che realizza l'interazione con l'utente proponendo ciclicamente i servizi che utilizzano i due metodi remoti, e stampa a video i risultati, fino alla fine del file di input da tastiera.

Parte C

Progettare un **servitore multiservizio (usando la select)** che consenta di effettuare le operazioni remote per:

- **contare il numero delle linee che iniziano con una lettera maiuscola e contengono almeno uno specificato numero di occorrenze di un carattere indicato all'interno di tutti i file di testo presenti sul direttorio remoto,**
- **trasferimento dal server al client di tutti i file di un direttorio** che iniziano con uno **specifico prefisso alfabetico** (max 4 caratteri alfabetici 'a'-'z' e 'A'-'Z').

utilizzando **un'unica connessione per ogni sessione cliente:** questo vincolo e' da intendersi come prioritario e fondamentale.

Più in dettaglio:

- Il **client_stream** è organizzato come un processo ciclico fino alla fine del file di input e realizza funzionalità di **trasferimento** dal server al client di **tutti i file di un direttorio che iniziano con uno specifico prefisso**. Per ogni richiesta, il client richiede all'utente e invia al server il *nome del direttorio* e il *prefisso*, quindi riceve dal server i *file di quel direttorio che iniziano col prefisso richiesto*, e li memorizza con i nomi e contenuto corretto nel proprio file system.
- Il **client_datagram** è organizzato come un processo ciclico fino alla fine del file di input e realizza la funzionalità di **conteggio numero delle linee che iniziano con una lettera maiuscola e contengono almeno uno specificato numero di occorrenze di un carattere indicato all'interno di tutti i file di testo presenti sul direttorio remoto**. Per ogni richiesta, il client richiede all'utente *un carattere maiuscolo* e il *numero di occorrenze*, quindi riceve dal server il numero di righe e lo stampa a video.
- Il **server** multiservizio discrimina le richieste utilizzando la primitiva select: il **server gestisce in modo parallelo** la funzionalità di trasferimento file dal server al client, mentre la funzionalità di conteggio occorrenze di una parola **e' realizzata in modo seriale**.
Per ogni richiesta di **trasferimento dal server al client**, il figlio riceve il nome del direttorio e il prefisso, e successivamente attua un protocollo per inviare al client tutti i file di testo che rispettano il criterio.
Per ogni richiesta di **conteggio del numero di linee** all'interno di **tutti i file di testo** presenti sul direttorio (corrente) remoto, il figlio legge il carattere e il numero di occorrenze, quindi esegue l'operazione di conteggio, scandendo ciascun file di testo riga per riga e contando il numero di righe che iniziando con una lettera maiuscola e contengono (in qualsiasi posizione della riga) almeno un numero di occorrenze del carattere pari (cioè maggiore o uguale) a quelle indicate dall'utente, su ciascun file di testo trovato; quindi invia la risposta al client, rappresentata da un intero positivo che indica il numero totale di occorrenze contate (≥ 0) in caso di successo, -1 in caso di problemi.