Reti di Calcolatori L-A
Appello del 12/12/2023
Compito 1

Tempo a disposizione: 3h

Cognome: .....
Nome: .....
Matricola: .....

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 vivamente 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**.
I file **testo sono file che contengono sempre solo linee** (con fine linea) composte di **caratteri stampabili di lunghezza massima specificata** e che hanno **nomi con il suffisso ".txt".**
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.

In particolare, si vogliono realizzare i seguenti servizi remoti su file testo:

1. **controllo dell'esistenza di almeno una linea con una parola specifica all'interno di un file testo**: l'operazione richiede all'utente un *nome di file* e una *parola*, controlla se il file richiesto contiene al suo interno almeno una linea contenente la parola cercata e restituisce *l'esito dell'operazione*;
2. **lista dei file di un direttorio che contengono al loro interno una linea con una parola specifica**: questa operazione richiede un *nome di direttorio* e una *parola*, quindi visualizza la *lista dei nomi di file* che si trovano nel direttorio richiesto e che contengono al loro interno almeno una linea con una o più un'occorrenze della parola cercata;
3. **conteggio del numero di linee di un file se il file contiene una linea con una parola specifica**: questa operazione richiede un *nome di file* e la *parola* e conta il *numero di linee* del file;
4. **trasferimento dal server al client delle linee di un file che contengono una parola specifica**: questa operazione richiede all'utente il *nome di un file di testo* e una *parola*, quindi legge il file a linee, e trasferisce dal server al client *tutte le linee* che contengono almeno una occorrenza della parola cercata, stampandola a video.

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)

Parte Java

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

- visualizzare la lista dei **nomi di file di un direttorio** che contengono al loro interno una linea con una parola specifica;
- contare il **numero di linee** di cui è costituito il file, se il file contiene **almeno una linea** con una parola specifica.

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_nomi_file_contenenti_parola_in_linea`** accetta come parametri d'ingresso il nome del direttorio e la parola cercata e restituisce la lista dei nomi di file che contengono al loro interno almeno una linea con una o più occorrenze della parola cercata.
- Il metodo **`conta_numero_linee`** accetta come parametro d'ingresso il nome del file e la parola e ne restituisce il numero di linee, 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 (uso di select) che consenta di effettuare le operazioni remote per:

- controllare l'esistenza di **almeno una linea** con **una parola specifica** all'interno **di un file**;
- trasferire dal server al client ***le linee di un file*** che contengono una **parola specifica**.

Più in dettaglio:

- Il **client_stream** è organizzato come un processo ciclico fino alla fine del file di input e realizza la funzionalità di **trasferimento delle linee** utilizzando **socket stream e un'unica socket e una unica connessione per sessione cliente**.
  Per ogni richiesta, il client richiede all'utente e invia al server il *nome del file* e la *parola cercata*, quindi riceve le *linee filtrate* e le stampa a video.
- Il **client_datagram** è organizzato come un processo ciclico fino alla fine del file di input e realizza la funzionalità di controllo dell'occorrenza di una linea con una parola utilizzando **socket datagram**.
  Per ogni richiesta, il client richiede all'utente e invia al server il *nome del file* e la *parola cercata*, quindi riceve l'esito e lo stampa a video.
- Il **server** principale unico discrimina le richieste utilizzando la primitiva **select**. Il server gestisce in modo parallelo la funzionalità di trasferimento linee dal server al client, mentre la funzionalità di controllo dell'occorrenza di una parola può essere realizzata in modo seriale o parallelo.
  Per ogni richiesta di **trasferimento di linee dal server al client**, il figlio riceve il *nome del file* e la *parola*, controlla che il file corrispondente esista sul server e trasmette al client *tutte le righe* che contengono al loro interno la parola ricercata, oppure notifica una indicazione di errore, ad esempio, se il file non esiste nel direttorio remoto.
  Per ogni richiesta di **controllo dell'occorrenza di una parola nel file**, il server riceve il *nome del file* e la *parola*, controlla se il file contiene al suo interno *almeno una linea* con una o più occorrenze della parola cercata e restituisce l'esito al client.