Reti di Calcolatori T
Appello del 14/06/2018
Compito 1

Cognome: ……
Nome: ……
Matricola: ……

Tempo a disposizione: 2h

È obbligatorio inserire Cognome, Nome, e numero di Matricola 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 sugli argomenti della richiesta e si gestiscano le eccezioni verso l'utente, 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 sul file system di una macchina server** (direttorio remoto), da parte di più clienti.
Ogni comando considera come direttorio di partenza il direttorio corrente, in cui siano stati lanciati rispettivamente il client e il server. Se per esempio il client richiede la cancellazione di un file (usando una notazione relativa) nel direttorio corrente, il server cerca di cancellare il file nel direttorio dove è stato lanciato.
I servizi considerano solo file di testo per realizzare le seguenti funzioni:

1. **lista dei file di un direttorio remoto i cui nomi contengano una sequenza di 2 o più vocali consecutive**: questa operazione richiede all'utente il nome del direttorio, quindi visualizza la lista dei file del direttorio e di ogni sottodirettorio il cui nome contiene una sequenza di due o più caratteri consecutivi alfabetici, vocali maiuscole e minuscole ('a', 'A', 'e', 'E', 'i', 'I', 'o', 'O', 'u', 'U');
2. **aggiunta di numerazione delle righe in un file di testo remoto:** questa operazione richiede all'utente il nome di un file, aggiunge all'inizio di ogni riga un intero progressivo, da 1 al numero di righe, e visualizza a video l'esito dell'operazione;
3. **eliminazione di tutte le occorrenze di caratteri consonanti all'interno di un file di testo remoto:** questa operazione richiede all'utente il nome di un file remoto ed elimina dal file tutte le occorrenze di caratteri alfabetici consonanti (diversi da 'a', 'A', 'e', 'E', 'i', 'I', 'o', 'O', 'u', 'U', maiuscole o minuscole), visualizzando a video il numero di eliminazioni effettuate;
4. **trasferimento di tutti file di un direttorio remoto i cui nomi contengono almeno un carattere alfabetico vocale e uno consonante (maiuscoli o minuscoli) dal server al client**: questa operazione richiede all'utente il nome del direttorio, e trasferisce i file di quel direttorio il cui nome contiene almeno un carattere alfabetico vocale e uno consonante salvandoli sul direttorio del client.

Si richiede inoltre di **non** usare nella soluzione 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:
- **ricevere la lista dei file di un direttorio remoto i cui nomi contengano una sequenza di 2 o più (caratteri alfabetici) vocali consecutive;**
- **aggiungere la numerazione delle righe in un file di testo remoto.**

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_file** accetta come parametri d'ingresso **il nome di un direttorio**, e restituisce la lista dei nomi dei file trovati con vocali consecutive. In caso di nessun file presente o di direttorio inesistente, prevedere una segnalazione di errore.
- Il metodo **numerazione_righe** accetta come parametro d'ingresso **il nome del file** e aggiunge all'inizio di ogni riga **un intero progressivo**, da 1 al numero di righe, per restituire un intero positivo con il numero di righe numerate (>=0) in caso di successo, oppure -1 in caso di errori o file inesistente.

Si progettino inoltre 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*), il processo 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 obbligatoriamente una select) che consenta di effettuare le operazioni remote per:
- **eliminare tutte le occorrenze di caratteri alfabetici consonanti all'interno di un file di testo remoto**,
- **trasferire tutti file di un direttorio remoto i cui nomi contengono almeno un carattere alfabetico vocale e uno consonante (maiuscoli o minuscoli) dal server al client,**

utilizzando un'**unica connessione per ogni sessione cliente**.
Più in dettaglio:
- Il **client_stream** è organizzato come un **processo filtro, ciclico fino alla fine del file di input** e che realizza la funzionalità di **trasferimento di tutti file di un direttorio remoto i cui nomi contengono almeno un carattere alfabetico vocale e uno consonante (maiuscoli o minuscoli) dal server al client** utilizzando **socket stream e un'unica socket**.
  Per ogni richiesta, il client chiede all'utente e invia al server il nome del direttorio, quindi riceve nome e contenuto di tutti i file e li salva sul file system locale.
- Il **client_datagram** è organizzato come un **processo filtro, ciclico fino alla fine del file di input** e che realizza la funzionalità di **eliminazione di tutte le occorrenze di caratteri consonanti all'interno di un file di testo remoto** utilizzando **socket datagram**.
  Per ogni richiesta, il client invia il nome del file e riceve l'esito del numero di eliminazioni stampandolo a video.
- Il **server principale unico** discrimina le richieste utilizzando la primitiva select: il **server gestisce in modo parallelo** la funzionalità di trasferimento di tutti file di un i cui nomi contengono almeno un carattere vocale e uno consonante dal server al client; mentre **realizza in modo o seriale o parallelo** la funzionalità di eliminazione di tutte le occorrenze di caratteri consonanti all'interno di un file di testo remoto: si motivi la scelta fatta.
  Per ogni richiesta di **trasferimento di tutti file di un direttorio i cui nomi contengono almeno un carattere alfabetico vocale e uno consonante (maiuscoli o minuscoli) dal server al client**, il figlio riceve il nome del direttorio, quindi invia al client tutti i file i cui nomi contengono almeno un carattere alfabetico vocale e uno consonante (maiuscoli o minuscoli) dal server al client.
  Per ogni richiesta di **eliminazione di tutte le occorrenze di caratteri consonanti all'interno di un file di testo remoto**, il server riceve il nome del file, quindi effettua l'operazione di eliminazione sul file locale e invia la risposta al client: un intero positivo che indica il numero di eliminazioni effettuate (>=0) in caso di successo, -1 in caso di problemi.