Reti di Calcolatori L-A
Reti di Calcolatori T
Appello del 21/01/2011
Compito 2

Cognome: ……
Nome: ……
Matricola: ……

Tempo a disposizione: 2h

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 sul file system di una macchina server** (direttorio remoto), **da parte di utenti che eseguono su 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 dei file nel direttorio corrente, il server conta i file presenti nel direttorio da cui è stato lanciato e spedisce al client la risposta.
In particolare si vogliono realizzare i seguenti servizi:

1. **lista dei sottodirettori il cui nome termina con un certo carattere minuscolo**: l'operazione richiede un nome di direttorio e visualizza la lista dei sottodirettori il cui nome termina con un carattere minuscolo;
2. **lista di tutti i file presenti nei sottodirettori di primo livello i cui nomi contengono due caratteri specificati**: questa operazione visualizza la lista di tutti i nomi di file contenuti nei sottodirettori di primo livello del direttorio corrente, per i sottodirettori il cui nome contiene entrambi i caratteri cercati;
3. **conteggio dei file testo di un direttorio che hanno un numero di linee maggiore di una certa soglia**: questa operazione richiede il nome di un direttorio e la soglia (un intero) e restituisce il numero di file di testo presenti nel direttorio che rispondono al requisito richiesto;
4. **trasferimento dal server al client di alcune linee selezionate di un file di testo**: questa operazione richiede all'utente il nome di un file di testo e una lunghezza di riga, quindi legge il file a linee, e trasferisce dal server al client le linee del file con lunghezza **maggiore** di quella indicata, stampandole a video.

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:
- visualizzare la lista dei sottodirettori i cui nomi terminano con un carattere minuscolo;
- visualizzare la lista di tutti i file presenti nei sottodirettori di primo livello i cui nomi contengono due caratteri.

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_sottodirettori_carattere`** accetta come parametro il nome del direttorio e restituisce la lista dei sottodirettori il cui nome termina con un carattere minuscolo.
- Il metodo **`lista_sottodirettori_due_caratteri`** accetta come parametri i due caratteri e restituisce la lista dei nomi di file di testo che si trovano nei sottodirettori il cui nome contiene entrambi i caratteri cercati.

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:
- trasferire dal server al client le righe di un file testo con **lunghezza maggiore** di quella specificata;
- contare i file di testo di un direttorio con un **numero di linee maggiore** di una soglia.

Più in dettaglio:
- Il **client_stream** è organizzato come un processo ciclico che legge fino alla fine del file di input e realizza la funzionalità di trasferimento dal server al client delle righe di un file di testo con lunghezza maggiore di quella richiesta utilizzando **socket stream e un'unica socket**.
  Per ogni richiesta, il client richiede all'utente e invia al server il nome del file di testo e la lunghezza di interesse, quindi riceve le righe del file inviate dal server e le stampa a video.

- Il **client_datagram** è organizzato come un processo ciclico che legge fino alla fine del file di input e realizza la funzionalità di conteggio dei file di testo di un direttorio che hanno un numero di linee maggiore della soglia specificata utilizzando **socket datagram**.
  Per ogni richiesta, il client invia al server il nome del direttorio e la soglia, quindi riceve la risposta e la stampa a video.

- Il **server** principale unico discrimina le richieste utilizzando la primitiva select. Il server gestisce in modo parallelo la funzionalità di trasferimento dal server al client delle righe di un file di testo con lunghezza maggiore di quella richiesta, mentre la funzionalità di conteggio dei file di testo di un direttorio che hanno un numero di linee maggiore di una soglia può essere realizzata in modo seriale o parallelo.
  Per ogni richiesta di **trasferimento dal server al client delle righe di un file di testo con una lunghezza maggiore di quella richiesta**, il figlio riceve il nome del file di testo e la lunghezza, controlla che il file corrispondente esista sul server, e trasmette al client le righe trovate, con lunghezza maggiore di quella richiesta, oppure notifica una indicazione di errore, ad esempio se il file non esiste.
  Per ogni richiesta di **conteggio dei file testo di un direttorio che hanno un numero di linee superiore ad una soglia**, il server riceve il nome del direttorio e la soglia, e trasmette al client il numero di file testo nel direttorio che hanno un numero di linee maggiore rispetto alla soglia cercata oppure una notifica di errore, ad esempio, se non è possibile aprire il sottodirettorio.