Esame

lunedì 3 marzo 2025 22:50

Composto di 3 parti tutte obbligatorie:

- Scritto
  – Test domande a risposta multipla
- Esercizio/Homework 1
- Esercizio/Homework 2

Esame Scritto

- Compito a quiz da fare direttamente al computer in laboratorio sulla piattaforma elearning.uniroma1.it. Domande a risposta chiusa; una sola opzione vera.
- Ci saranno un certo numero di domande da fare in poco tempo!! 30 minuti per 30/40 domande (dettagli verranno forniti prima dell'esame).
- Nel caso in cui ci siano piu iscritti all'esame che posti in laboratorio, l'esame si fara in piu turni, comunicati sul sito del corso
- Voto in trentesimi

Homework (1 e 2)

- Da svolgere in gruppo 2 - 3 studenti (gruppi singoli ammessi solo per studenti con status lavoratore)
- Nel programma del corso sono allocate 9 ore per homework 1 e 12 ore per homework 2
- Da consegnare prima dell'esame scritto (vedi scadenze). Dovra' essere consegnato codice e relazione (dettagli verranno forniti con il testo degli homework)
- verificato mediante presentazione di tutti i membri del gruppo
- voto in trentesimi (un voto per ogni homework)
  Voto Finale = MEDIA(Voto Scritto + Voto Homework 1 + Voto Homework 2)

Gli homework vanno consegnati prima di svolgere l'esame scritto.
Le scadenze per la consegna degli homework sono le seguenti:

- 3/06/2024 (21:00) per svolgere l'esame il 9-10/06/2024
- 4/07/2024 (21:00) per svolgere l'esame il 11-14/07/2024
- 2/09/2024 per svolgere l'esame il 9/09/2024
- **in tutte le sessioni a seguire (aka Sessione invernale e Sessioni straordinarie**) 5 giorni lavorativi antecedenti la data di esame

Regole Verbalizzazione Esami

- Secondo Modulo SO (SOII) non va verbalizzato direttamente ma confluisce in un'unica verbalizzazione con il Primo Modulo (SOI). SOI + SOII = SO (12 crediti)
- Per canale M-Z verbalizza Prof. FABIO DE GASPARI - notifico io il superamento di SOII. Non dovete mandare voi studenti mai al prof. De Gasperi.
- if (Voto_SOI >= 18 AND Voto_SOII >= 18) **Voto_SO** = (Voto_SOI + Voto_SOII) / 2;
- **Voti SOI e SOII devono essere conseguiti nell'arco temporale di un anno solare dal superamento del primo dei due moduli.**

MyFileTransferApp

venerdì 21 febbraio 2025 15:25

Lo studente deve realizzare un applicazione client-server che permette di trasferire file da un client and un server (scrittura di un file) e, viceversa, da un server ad un client (lettura di un file).

L'applicazione è composta da due programmi, un client che effettua richieste di lettura e scrittura di un file, un server che riceve le richieste e gestisce le richieste dei client.

Assumendo che l'applicazione server si chiama myFTserver e l'applicazione client si chiama myFTclient il comportamento deve essere il seguente.

Server

Il comando
myFTserver -a server_address -p server_port -d ft_root_directory

esegue il programma server mettendolo in ascolto su di un determinato indirizzo IP e porta ed indicando la directory nella quale andare a scrivere/leggere i file. Se ft_root_directory non esiste deve essere creata.

Una volta in esecuzione, il server deve accettare connessioni da uno o piu' client e gestirle concorrentemente.
Richieste di scrittura concorrenti sullo stesso file devono essere opportunamente gestite (come la richiesta di creazione concorrente di path con lo stesso nome).
Il programma server deve gestire tutte le eccezioni come ad esempio: richiesta di accesso a file non esistente (per la lettura), errore nel binding su IP e porta, parametri di invocazione del comando errati o mancanti, spazio su disco esaurito, interruzione della connessione con il client.

Client

Il comando
myFTclient -w -a server_address -p port -f local_path/filename_local -o remote_path/filename_remote

esegue il programma client, crea una connessione con il server specificato da server_address:port, e scrive il file local_path/filename_local sul server con nome filename_remote e nella directory specificata da remote_path. remote_path avra' root nella directory del server specificata con ft_root_directory

il comando
myFTclient -w -a server_address -p port -f local_path/filename_local

si comporta come il precedente ma il nome del path remoto e del file remoto sono gli stessi del path e file locale.

il comando
myFTclient -r -a server_address -p port -f remote_path/filename_remote -o local_path/filename_local

esegue il programma client, crea una connessione con il server specificato da server_address:port e legge il file specficato da remote_path/filename_remote trasferendolo al programma client che lo scrivera' nella directory local_path assegnando il nome filename_local

il comando
myFTclient -r -a server_address -p port -f remote_path/filename_remote

si comporta come il precedente ma il nome del path locale e del file locale sono gli stessi del path e file remoto.

il comando
myFTclient -l -a server_address -p port -f remote_path/

permette al client di ottenere la lista dei file che si trovano in remote_path (effettua sostanzialmente un ls -la remoto). la lista dei file deve essere visualizzata sullo standard output del terminale da cui viene eseguito il programma myFTclient.

Il programma client deve gestire tutte le eccezioni del caso. come ad esempio: parametri di input errati, file remoto non esistente (lettura), spazio di archiviazione insufficiente sul server (scrittura) e sul client, interruzione della connessione con il server

Valutazione:

Il codice sottomesso deve essere commentato in modo chiaro ed estensivo
il codice deve essere accompagnato da una relazione (2 pagine massimo) che descriva le scelte operate.
In sede di verifica, la studentessa/lo studente deve dimostrare che il codice funzioni

La consegna deve avvenire come segue:

Sottomettere un archivio .tgz o .zip, nominato come MATRICOLA_HW2 contenente i file .h, .c e .pdf (per la relazione)

MyFinger

venerdì 21 febbraio 2025 15:30

L'homework consiste nel creare un programma che ha lo stesso comportamento del comando finger[1]

Non e' richiesta l'implementazione del lookup delle informazioni per utenti remoti
Nell'implementazione non puo' essere utilizzata la famiglia di funzioni exec*(3)

Il codice deve essere organizzato in almeno un file header (.h) che contiene tutte le dichiarazioni di variabili, costanti, e prototipi di funzioni) ed un file .c che contiene il codice delle funzioni progettate ed implementate.

Valutazione:

Il codice sottomesso deve essere commentato in modo chiaro ed estensivo
il codice deve essere accompagnato da una relazione (2 pagine massimo) che descriva le scelte operate.
In sede di verifica, la studentessa/lo studente deve dimostrare che il codice funzioni

La consegna deve avvenire come segue:

Sottomettere un archivio .tgz o .zip, nominato come MATRICOLA_HW1 contenente i file .h, .c e .pdf (per la relazione)

Riferimenti

[1] finger(1) - Linux man page https://linux.die.net/man/1/finger