chat-room SO2

- anno 2020/21
- progetto di marco wang

componenti

server

- il server ci permettere di avviare un server, possiamo modificare due parametri all'avvio: la `porta` e la `modalita'` (`ricezione:-r, timestamp client: -t`), non importa l'ordine
- per avviare basta scrivere `./server`, se non si passano i parametri verra' avviato con i parametri di default che sono porta: 9000 e modalita': ricezione del server(fifo)
- per altre modificare altri parametri meno frequenti e' possibile andare nel file defaultServer.h
- utilizza tre struttura principali per gestire la `coda`, i `messaggi`, e i `client`
- utilizza N thread per gestire N utenti, un thread per l'invio di messaggi(ed eventuale riordino), un thread per eseguire i comandi

client

- il client ci permette di avviare un client, possiamo modificare il numero di porta all'avvio
- permette di comunicare con il server utilizzando le socket
- nel main manda i messaggi, e affida ad un thread la ricezione

Makefile

- permette la compilazione mediante il comando make possiamo ottenere un eseguibile client e un server mediante i comandi `make client`, `make server` oppure tutti e due mediante un solo comando: `make all` -possiamo pulire i file oggetto con `make clean`

per eseguire (parametri non obbligatori)

```
 - make
- ./server [mode -t,-r] [porta: int]
- ./client [porta: int]
```

---

struttura del progetto

- ## `src` : sono presenti tutti i file sorgenti
  - ### `client` : sono presenti tutti i file necessari a creare il client
    - `header` : ci sono i corrispettivi header dei file .c
    - `client.c` : file principale del client
    - `defaultClient.c` : funzioni ausiliari al client e impostazioni di default
  - ### `server` : sono presenti tutti i file necessari a creare il server
    - `header` : ci sono i corrispettivi header dei file .c
      - `defaultServer.h` : header file che contiene le impostazioni di default come porta/modalita' e lunghezza messaggi
    - `server.c` file principale del server dove vengono implementate le funzioni di invio e ricezione
    - `defaultServer.c` funzioni ausiliari al server
    - `structClient.c` struttura che il server usa per gestire i client, ha una radice e forma una catena
    - `structQueue.c` struttura che il server usa per gestire i messaggi, e' una coda circolare
  - `default.c` : funzioni ausiliare sia al client che al server
- `Makefile` : permette di compilare con il comando make
  - all : compila tutti i file e genera (client e server)
  - client : compila i file della cartella client+ default.c e genera client
  - server : compila i file della cartella server+ default.c e genera server
  - clean : rimuove i file .o generate nelle varie cartelle
- `logFile` *questa cartella verra' create all'avvio del server la prima volta e serve per mantenere i file di log, nelle successive volte verra' creatoc da data/..*
  - `data` :cartella che contiene i file di log di quel giorno formato YY:MM:DD
    - `clients` :contiene tutti i client che si sono conessi in questo giorno
      - `client1.txt`
      - `client2.txt`

        - ....
    - `serverLog.txt`
- ogni giorno che verra' avviato il server si creera' una gerarchia di cartelle a partire da `data`

test

- ### piu' client possono unirsi ad un server
    - basta avviare il server(./server) e avviare diversi client con la stessa porta!
- ### un messaggio inviato, viene visto da tutti
    - avviamo il server e almeno due client, dopodiche' inserire il nome nei client inviamo il messaggio!
    - ### due modalita di distrubuzione:
    - `ricezione(fifo)` : basta avviare il server facendo `./server -r` , e tutti i messaggi inviato dagli utenti verranno immagazzinati nella coda, e un thread inviera' a seconda dell'ordine di arrivo
    - `timestamp` : avviamo il server facendo `./server -t` , tutti i messaggi vengono imagazzinati nella coda e il thread restera' N secondi in sleep per permettere l'arrivo di messaggi dopodiche' riordina e avra altri N1 secondi per inviare i messaggi
    - ### log dei client e del server:
    - al primo avvio creera' tre cartelle, sucessivamente solo due una del giorno corrente e all'interno quella dei client