**Relazione Progetto**

Traccia 3 – CHATGAME

Fabbri Leroy 0000933532

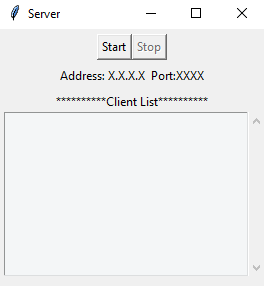
Magrini Giacomo 0000933574

1. Introduzione

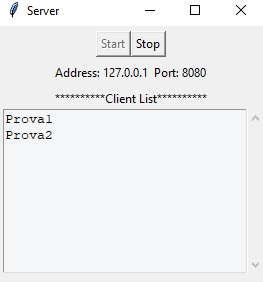
Il progetto consiste nella realizzazione di un Chatgame con due giocatori attraverso l’utilizzo del linguaggio Python. Per realizzare ciò si avrà una connessione tra client e server sfuttando il protocollo TCP-IP.

1. Descrizione

Per avviare il progetto, bisogna creare 3 console distinte (una per il server e due per i client). All’avvio della classe “game\_server.py” cliccare su “Start” per avviare il server, comparirà di seguito l’indirizzo a cui i client dovranno connettersi.



Successivamente, inserire un nome su ogni client e cliccare connetti per iniziare a giocare. Il gioco non partirà a meno che non siano presenti tutti e due i giocatori, nella tab del server compariranno i giocatori connessi.



Raggiunto il numero di giocatori, il client si espanderà, rivelando il nome del giocatore, un ruolo assegnato dal server, il tempo rimanente alla fine della partita e tre caselle cliccabili. Una delle tre nasconderà un trabocchetto che concluderà direttamente la partita.



Dopo aver interagito con uno dei tre pulsanti, se non è stato selezionato il trabocchetto, comparirà una domanda e le relative risposte, disabilitando i tre pulsanti.



Dopo aver risposto, la domanda e le risposte scompariranno, riabilitando i tre pulsanti precedenti. Il gioco prosegue fino allo scadere del tempo o finchè un giocatore non seleziona il trabocchetto.

Se alla selezione della domanda entrambi i giocatori selezionano il trabocchetto, comparirà un messaggio di pareggio, altrimenti verrà assegnata la vittoria direttamente al giocatore che ha evitato la trappola.

Allo scadere del tempo, il giocatore che ha risposto correttamente a più domande è il vincitore della partita.



Al termine della partita, per qualsiasi situazione, comparirà un timer che mostra il tempo alla chiusura della connessione del client con il server e del programma.

1. Dettagli implementativi

I threads utilizzati sono i seguenti:

Al lato server avremo un thread che servirà per accettare i client e riceverne i messaggi.

Al lato client invece saranno utilizzati tre threads. Il thread principale servirà per ricevere messaggi dal server, un thread che utilizziamo per il tempo della partita, e l’ultimo che utilizziamo per il tempo della chiusura del programma, dopo la fine della partita.

Le strutture dati utilizzate sono:

Due dizionari, che abbiamo utilizzato per memorizzare le domande e le risposte;

Un set, che abbiamo utilizzato per memorizzare solo le risposte corrette;

Un set, in cui sono stati inseriti i ruoli che il server da all’utente;

Una lista di client connessi al server;

Una lista con i nomi dei client;

Una lista con i dati che il client invia al server

Una volta che i due client sono connessi al server e il gioco è cominciato, la ricezione di messaggi tra server e client avviene nel seguente modo:

Il client invia al server la stringa del pulsante che ha cliccato che ritiene essere la risposta corretta. Questa stringa verrà ricevuta e a sua volta inviata all’altro client, la quale verrà decodificata e assegnata ad una variabile. Dopodichè parte la funzione “logic” che avrà come parametro due stringhe: una è “tua\_scelta”, che è semplicemente la risposta che l’utente ritiene essere corretta, mentre “scelta\_avversario” è la risposta dell’avversario che il client ha ricevuto tramite il server.

Nella “logic” si controlla che la propria risposta e quella dell’avversario siano presenti nel set delle risposte corrette. Nel caso sia corretta, si aumenta di 1 il punteggio, altrimenti il punteggio verrà diminuito, sempre di 1.

In questa maniera viene gestito anche il caso in cui si seleziona il trabocchetto, perché al posto di inviare una stringa di risposta, verrà inviata una stringa speciale inizializzata come campo. La “logic” controlla che “scelta\_avversario” e “tua\_scelta” non contengano la stringa speciale, e assegna il risultato della partita a seconda di chi ha premuto il pulsante trabocchetto, facendo partire il timer della chiusura della partita e interrompendo la comunicazione con il server.

1. librerie utilizzate e classi sviluppate

Librerie:

tkinter

messagebox from tkinter

socket

sleep from time

threading

random

Classi:

chatgame\_client.py

chatgame\_server.py

1. risposte alle domande a fini di debug

1 : "Su che pianeta è ambientato Terraformars" Marte

2 : "Chi è il protagonista di DragonBall " Goku

3 : "In che guerra è ambientato battlefield 1" Prima Guerra Mondiale

4 : "In che Continente è avvenuta la guerra di Ace Combat 04 " USEA

5 : "Ultimo capitolo di Metro" Metro Exodus

6 : "Quale dei seguenti videogiochi NON e della Paradox" Total War: Empire

7 : "Come si chiama il protagonista di Red Dead Redemption" John Marston

8 : "quanto è stato rilasciato per la prima volta Warframe" 2013

9 : "Gioco dell'anno del 2018" Fortnite

10 : "Come si chiama il remake di Nier: Gestalt" Nier Replicant

11: "Che arma usa Sinon di GGO" Fucile di precisione

12: "Come si chiama la VTuber drago di Hololive" Kiryu Coco

13: "Protagonista femminile di Steins;Gate" Kurisu Makise

14: "Quanti anni ha Uzaki-chan all'inizio della stagione" 19

15: "Miglior anime 2019" Attack on Titan S4