Laboratorio per il corso di Reti I (A.A. 2016/17) – Applicazione A

Sviluppare in linguaggio C un'applicazione di rete, costituita da un programma server ed un programma client, in cui l'utente del programma client deve indovinare un numero scelto casualmente dal programma server, secondo le specifiche sotto indicate.

SERVER

1. Il programma server viene eseguito indicando sulla riga di comando la porta sulla quale mettersi in ascolto
   `$ programma_server <numero porta>`
2. All'apertura della connessione il server sceglie un numero casuale tra 1 e 100
3. il server manda quindi un messaggio di benvenuto, nel formato
   `OK <Messaggio>`
   ovvero la stringa '`OK`', seguita da uno spazio e da una stringa personalizzabile dal server. Il messaggio deve avere una lunghezza massima di 256 caratteri ed è terminato da un carattere di andata a capo ('`\n`')
4. Il server si pone in attesa di un messaggio da parte del client. Il messaggio deve essere costituito da una stringa di testo nel formato
   `<Numero>`
   ovvero un numero rappresentato come una stringa di caratteri ASCII (ad esempio '`5`', '`41`', '`83`'), terminata da un carattere di andata a capo ('`\n`')
5. Il server confronta il valore numerico ricevuto con il numero casuale prescelto:
   a. Se è minore, risponde con il messaggio
      `NO +`
      ovvero la stringa '`NO`' seguita da uno spazio e dal carattere '+' e terminata da un carattere di andata a capo ('`\n`').
   b. Se è maggiore, risponde con il messaggio
      `NO -`
      ovvero la stringa '`NO`' seguita da uno spazio e dal carattere '–' e terminata da un carattere di andata a capo ('`\n`').
   c. Se è uguale, risponde con il messaggio
      `SI <Messaggio>`
      ovvero la stringa '`SI`', seguita da uno spazio e da una stringa personalizzabile dal server. Il messaggio deve avere una lunghezza massima di 256 caratteri ed è terminato da un carattere di andata a capo ('`\n`')
   d. Se il valore numerico ricevuto è minore di 1 o maggiore di 100, oppure non ha una rappresentazione numerica valida, risponde con il messaggio
      `ER <Messaggio>`
      ovvero la stringa '`ER`', seguita da uno spazio e da una stringa personalizzabile dal server. Il messaggio deve avere una lunghezza massima di 256 caratteri ed è terminato da un carattere di andata a capo ('`\n`')
6. Se il client non ha indovinato il numero prescelto (casi 5a e 5b), il server ritorna al punto 4 in attesa di un nuovo messaggio del client
7. Se il client ha indovinato il numero prescelto (caso 5c), il server chiude la connessione e si pone in attesa della richiesta di un nuovo client
8. Se il client ha mandato un messaggio dal formato non valido (caso 5d), il server chiude la connessione e si pone in attesa della richiesta di un nuovo client

9. (Opzionale) Il server può decidere di porre un limite al numero di tentativi effettuati dal client. In questo caso, il server conta i tentativi del client e se il loro numero supera la soglia prefissata risponde con un messaggio di errore "`ER <messaggio>`", comportandosi come nel caso 5d, ovvero chiudendo la connessione e ponendosi in attesa della richiesta di un nuovo client.

CLIENT

1. Il programma client viene eseguito indicando sulla riga di comando l'indirizzo IPv4 del server da contattare e la porta sulla quale contattarlo
   `$ programma_client <indirizzo_server> <numero_porta>`
2. Dopo l'apertura della connessione il client si aspetta di ricevere dal server il messaggio di benvenuto, nel formato
   `OK <Messaggio>`
   ovvero la stringa '`OK`', seguita da uno spazio e da una stringa personalizzata dal server. Il messaggio deve avere una lunghezza massima di 256 caratteri ed è terminato da un carattere di andata a capo ('`\n`')
3. Il client presenta all'utente il messaggio del server, senza il delimitatore "`OK `" che costituisce una parola chiave del protocollo di scambio e non fa parte del messaggio del server
4. Il client sollecita l'utente ad inserire, da tastiera, un numero intero tra 1 e 100
5. Il client spedisce tale numero al server in un messaggio costituito da una stringa di testo nel formato
   `<Numero>`
   ovvero con il numero rappresentato come una stringa di caratteri ASCII (ad esempio '`5`', '`41`', '`83`'), terminata da un carattere di andata a capo ('`\n`')
6. Il client si pone in attesa di una risposta da parte del server. Le risposte possibili sono:
   a. `NO –`
   b. `NO +`
   c. `SI <Messaggio>`
   d. `ER <Messaggio>`
7. Se la risposta indica un tentativo errato (casi 6a e 6b), il client lo comunica all'utente con un opportuno messaggio e torna al punto 4 a chiedere all'utente di inserire un numero
8. Se la risposta indica successo (caso 6c), il client lo comunica all'utente riportando il messaggio del server (senza il delimitatore "`SI `"), insieme ad un eventuale messaggio da parte del client, dopo di che chiude la connessione e termina l'esecuzione
9. Se la risposta indica un errore (caso 6d), il client lo comunica all'utente riportando il messaggio del server (senza il delimitatore "`ER `"), insieme ad un eventuale messaggio da parte del client, dopo di che chiude la connessione e termina l'esecuzione

Entrambi i programmi devono gestire opportunamente tutti i possibili errori che si possono verificare in fase di apertura e chiusura delle connessioni.
Tutti i messaggi scambiati sono costituiti da stringhe terminate da un carattere di andata a capo ('`\n`').
Entrambi i programmi client e server devono essere in grado in inter-operare con programmi analoghi che rispettino i requisiti elencati: per questo motivo i messaggi di rete devono rispettare rigorosamente i formati definiti. L'interoperabilità è un requisito fondamentale che verrà

verificato in fase di valutazione. Ogni altro aspetto di interazione tra il client, il server e l'utente, può essere personalizzato a piacimento, ad esempio i messaggi a schermo di notifica o di errore.

Esempi di comunicazione sulla rete

```
S: OK Benvenuto, indovina che numero ho pensato
C: 20
S: NO +
C: 50
S: NO –
C: 42
S: SI Bravo, hai indovinato!


S: OK Benvenuto, indovina che numero ho pensato. Hai 10 tentativi
C: 20
S: NO +
C: 101
S: ER Hai sbagliato! 101 non e' una risposta valida. Arrivederci
```