Prova in itinere 05/12/2018
Socket (in C o Java) -- Tempo a disposizione: 60 minuti
N.B.: Consegnare tutti i file sorgente prodotti, denominati secondo lo schema: `server_A.c` o `Server_A.java`, `server_B.c` o `Server_B.java`...

A. Realizzare un server (in C o in Java) che si metta in ascolto sul **port 3333**, sul quale accetta richieste di connessione da un client.
   Sulla connessione stabilita, il server riceve una stringa**,** terminata dal carattere **'\n'** (si supponga che basti una sola operazione di lettura in ricezione da parte del server, per ricevere ciascuna stringa inviata dal client).
   Il server quindi scrive la stringa ricevuta sullo standard output e si rimette in attesa di un'altra eventuale stringa, o, se il cliente chiude la connessione, si rimette in attesa di richieste di connessione.
   Testare il server *server_A* usando *telnet*.

B. Conservando la versione (A) del server, realizzarne un'ulteriore versione tale che ciascuna stringa ricevuta dal server, oltre ad essere scritta dal server sulla propria standard output, sia da esso inviata nuovamente al client come risposta (senza alcuna modifica).
   Testare il server *server_B* usando *telnet*.

C. Conservando la versione (B) del server, realizzarne un'ulteriore versione tale che il server implementi il metodo/funzione

   ```
   String list(s);
   ```

   che, **per il momento, ai fini di questo quesito,** restituisce la stringa `s` stessa.
   Per ciascuna stringa `s` ricevuta dal cliente, il server invoca su di essa `list(s)` e invia in risposta al cliente la stringa restituita dalla funzione.
   Testare il server *server_C* usando telnet.

D. Supposto che il server mantenga come parte del proprio stato una stringa *history*, inizialmente vuota, Modificare la funzione/metodo `list(s)` in modo che:

   a. se `s` è diversa da `"SHOW"`, `list()` appenda `s` in coda a *history* usando "*;*" come carattere separatore, restituendo la stringa `"OK"` come risultato. Nota: *history* è unica per tutti i client e viene mantenuta tra una connessione e l'altra, finché il server non termina la sua esecuzione;

   b. se `s` è uguale a `"SHOW"`, `list()` restituisca la stringa *history*.

E. (Opzionale) Realizzare un semplice client per comunicare con i server creati ai punti precedenti.

Ciascun quesito va affrontato solo dopo aver risolto il precedente, producendo del codice funzionante.

Archiviare i file sorgente prodotti, `server_A.c` o `Server_A.java`, … etc. in un file compresso denominato `nome_cognome_matricola.zip` e caricarli entro 60 minuti all'indirizzo:

https://www.dropbox.com/request/4dPzmK2X1x5vLZzJg4mc

Al termine dei 60 minuti non sara' piu' possibile caricare ulteriori files mediante il link

Thread (sia in C che in Java) -- Tempo a disposizione: 60 minuti
N.B.: Consegnare tutti i file sorgente prodotti (C e Java).

Scrivere in C e in Java un programma con due thread *TP* e *TD* che condividono una variabile `m` intera, che va inizializzata a 0.

I thread eseguono un ciclo infinito, comportandosi rispettivamente come segue:

TP attende 300 ms e successivamente controlla il valore *M* di `m`:

- se *M* è pari:
  - scrive *M* sulla standard output
  - genera un numero dispari casuale compreso tra 0 e 9 e lo memorizza in *M*
  - TP sveglia *TD*
- se invece *M* è dispari, *TP* si mette in attesa.

TD attende 300 ms e successivamente controlla il valore di M:

- se *M* è dispari:
  - scrive *M* sulla standard output,
  - genera un numero casuale compreso tra 0 e 9 e lo memorizza in *M;*
  - *TD* sveglia *TP*
- Se invece *M* è pari, *TP* si mette in attesa.

N.B.

- in C, la chiamata `usleep(ut)` attende per `ut` microsecondi; in Java il metodo `sleep(mt)` della classe `Thread` attende per mt millisecondi.
- In C, ciascun thread esegue una funzione diversa `fun_p()` per TP e `fun_d()` per TD; in Java, ciascun thread è istanza di una classe diversa, `ThreadP` per `TP` e `ThreadD` per `TD`.

Archiviare i file sorgente in un file denominato nome_cognome_matricola.zip e caricarli entro 60 minuti all'indirizzo:

https://goo.gl/...

Al termine dei 60 minuti non sara' più possibile caricare ulteriori files.