Esercizio 6.1

Scrivere una classe di thread (`Printer`) in Java in cui il metodo `run()` realizzi la scrittura a video ripetuta per 100 volte di una stringa composta dall'indice di iterazione (del ciclo) e da una messaggio memorizzato in una variabile di istanza, inizializzata dal costruttore. Scrivere un programma che testi il funzionamento della classe istanziando tre thread di tipo `Printer`, ciascuno caratterizzato da un messaggio diverso. Utilizzare i due approcci visti a lezione per la definizione del corpo dei thread: (a) estensione della classe Thread; (b) implementazione dell'interfaccia Runnable.

Esercizio 6.2

Scrivere un'applicazione client/server in Java con socket orientate ai flussi (protocollo di trasporto: TCP) che consenta di effettuare in remoto il fattoriale di un numero intero. Il client dovrà leggere da stdin il numero di cui calcolare il fattoriale e inviarlo al server. Questo, una volta ricevuto il numero, ne calcola il fattoriale restituendo il valore ottenuto al client per la visualizzazione.
Scrivere una variante del server in modo da conferirgli un **comportamento concorrente**: ogni client sarà servito da uno specifico thread attivato dal server.

Esercizio 6.3

Implementare una variante del client dell'esercizio 6.2 in modo da emulare il comportamento di un numero specificato di client concorrenti che effettuano contemporaneamente richieste per il calcolo del fattoriale di numeri interi allo stesso server. Il programma client, che in questo caso chiameremo FactorialTCPClients, dovrà prevedere un thread di controllo per emulare il comportamento di ciascun client, la cui classe chiameremo FactorialTCPClientThread.
Usare questa variante del client per testare sia l'implementazione sequenziale sia quella concorrente dell'esercizio 6.2. E' possibile verificare quali e quanti thread sono attivati sul server aggiungendo nell'handler la stampa sullo stdout del nome del thread corrente.

Esercizio 6.4

Usando i thread in Java, realizzare la versione full-duplex dell'esercizio 4.3 (Chat Half Duplex). Si tenga presente che il codice che riguarda la lettura da stdin e la scrittura sul canale di rete e quello che riguarda la lettura dal canale di rete e la scrittura su stdout deve essere eseguito in modo concorrente al fine di consentire di inviare e ricevere messaggi in qualsiasi momento. Si implementino allo scopo due classi di thread: **`Writer`** e **`Reader`**.