Università di Ferrara
Laurea Triennale in Informatica
A.A. 2021-2022
Sistemi Operativi e Laboratorio

Lab-09. I Thread in Java

Prof. Carlo Giannelli

http://www.unife.it/scienze/informatica/insegnamenti/sistemi-operativi-laboratorio
http://docente.unife.it/carlo.giannelli
https://ds.unife.it/people/carlo.giannelli

Esercizio 5: 1/2

Si realizzi un programma Java multithread che realizza un termostato intelligente per un ambiente smart-home. Per realizzare tale sistema, il programma dovrà definire 2 thread «Sensor» e «Actuator».

Il thread «**Sensor**» simula il comportamento di un sensore di temperatura e ogni 300 ms genera un numero casuale in virgola mobile compreso tra 18.0 e 21.0 (temperatura in gradi Celsius). Generato il valore di temperatura «Sensor» invia tale dato al thread «Actuator».

Il thread «**Actuator**» si occupa di analizzare il valore ricevuto e di stampare a video un messaggio del tipo «Accendi riscaldamento, valore di temperatura corrente: 18.3» qualora tale valore sia inferiore a un valore soglia fornito in input dall'utente.

Esercizio 5: 2/2

Il main-thread deve occuparsi di chiedere all’utente il valore soglia desiderato e di creare/avviare i thread «Sensor» e «Actuator».

Realizzare la comunicazione inter-thread con le 3 modalità di de/serializzazione viste: 1) stream, 2) buffered, 3) oggetti

Esercizio 5: traccia

- Definire un metodo main che chiede all'utente il valore soglia desiderato
- Definire nel metodo main un canale di comunicazione (come visto nell'esercizio 2) per lo scambio di messaggi fra i thread Sensor e Actuator
- Definire un thread Sensor che si occupa di generare un numero casuale compreso tra 18.0 e 21.0 ogni 300 ms, ad esempio:

```
// 18 + random.nextFloat() * (21 - 18);
```

  Sensor invia tale valore ad Actuator utilizzando il canale di comunicazione creato nel main
- Definire un thread Actuator che legge il messaggio inviatogli da Sensor e stampa il messaggio «Accendere il riscaldamento» se la temperatura è inferiore al valore richiesto dall'utente
- Infine, il metodo main deve creare ed eseguire un'istanza di Sensor e Actuator

Esercizio 6:

Si realizzi un'applicazione Java multithread e **thread-safe** che simuli in maniera sommaria il comportamento di un activity manager di un sistema operativo. A tal fine, il programma dovrà creare 3 thread che chiameremo "Monitor", "Sorter" e "Manager".

Il thread "**Monitor**" dovrà registrare ogni 200 ms le informazioni del consumo di risorse (CPU) di un thread "ipotetico" all'interno di una struttura dati condivisa "**ThreadLoad**" tra "Monitor" e "Sorter". Per implementare tale funzionalità si generi un numero casuale da 0 a 9 rappresentante il thread-id e un numero casuale da 0 a 100 rappresentante il CPU load attuale.

Il thread "**Sorter**" si dovrà invece occupare di controllare ogni 150 ms quale sia il thread con il più alto CPU load. Ottenuto tale dato, "Sorter" si occupa di preparare un messaggio contenente l'id del thread e il CPU load e invia tale informazione a "**Manager**", che si occuperà di stampare a video il messaggio ricevuto. Infine, il Thread "Manager" si deve occupare di terminare "Monitor" e "Sorter" dopo aver stampato per dieci volte tale informazione.

Esercizio 6: traccia (1/3)

- Definire una classe thread-safe ThreadLoad in grado di memorizzare una serie di coppie <intero, double>: l'intero rappresenta l'id di un thread, il double il suo CpuLoad
- Un'istanza di threadLoad sarà condivisa tra i thread Monitor e Sorter: attenzione a implementare una corretta gestione della concorrenza!
- Definire un thread Manager che si dovrà occupare di 1) ricevere un messaggio dal thread Sorter, 2) stampare il messaggio ricevuto a video, 3) terminare i thread Monitor e Sorter dopo avere ricevuto 10 messaggi da Sorter

Esercizio 6: traccia (2/3)

- Definire un thread Monitor che ogni 200 ms memorizzi una coppia casuale (rappresentate il consumo di CPU di un thread) all'interno dell'istanza condivisa della classe ThreadLoad.
- Definire un thread Sorter che viene inizializzato con il riferimento all'istanza condivisa di ThreadLoad; ogni 150ms Sorter seleziona il thread registrato in ThreadLoad che ha registrato il più alto consumo di CPU e invia tale informazione al thread Manager (utilizzando uno dei metodi visti per la comunicazione fra Thread).
- Definire il metodo main che crea un'istanza di ThreadLoad, le istanze dei thread richiesti e le manda in esecuzione.

Esercizio 6: traccia (3/3)

- In sintesi:
  - Monitor scrive su ThreadLoad
  - Sorter legge da ThreadLoad e invia messaggi a Manager
  - Manager riceve messaggi da Sorter