L'obbiettivo di questa serie di esercizi è di darvi la possibilità di familiarizzare con gli strumenti messi a disposizione da Java per l'esecuzione dei threads.

Esercizio 1

Scaricate da iCorsi il codice sorgente S1Esercizio1.java ed analizzatelo nel dettaglio. Provate a compilarlo ed eseguirlo. In seguito, ristrutturate il programma e sviluppatene 3 varianti con approcci alternativi a quello dell'anonymous inner class: una in cui il thread estenda direttamente la classe Thread, una in cui venga estesa l'interfaccia Runnable e una in cui venga utilizzata una lambda expression. Assicuratevi che il programma mantenga sempre il medesimo comportamento.

Esercizio 2

Sviluppate un programma che crei 2 threads (implementando l'interfaccia Runnable di Java) che, quale unico comportamento, si mettano a dormire (utilizzando il comando Thread.sleep()) per un periodo di tempo casuale (tra 1500 e 2000 ms) e che, in seguito, stampino a schermo per quanto tempo sono stati inattivi. Importante: i tempi d'inattività dei threads dovranno venir calcolati dal thread principale del programma e forniti ai 2 threads al momento di costruzione. Prima che il programma termini, il thread principale dovrà assicurarsi che entrambi i threads abbiano completato l'esecuzione. Il programma dovrà avere un output simile al seguente:

```
Partono tutti i threads.
In attesa che i threads abbiano terminato.
Thread 1 risveglio dopo 1612 ms
Thread 0 risveglio dopo 1928 ms
Tutti i threads hanno terminato.
```

Esercizio 3

Sviluppate un programma che calcoli in parallelo la somma parziale dei valori contenuti in un array di 10000 elementi. Il thread principale dovrà occuparsi di inizializzare l'array con numeri casuali tra 1 e 100, creare 10 threads a cui assegnare a construction time l'array e l'intervallo sul quale operare, far partire i threads ed attendere che i threads abbiano terminato l'esecuzione. Ogni thread dovrà occuparsi di sommare gli elementi dell'array nell'intervallo a lui assegnato e, successivamente, stampare a schermo la somma ottenuta. Il primo thread dovrà quindi sommare i primi 1000 elementi, il secondo i 1000 successivi, e così via.
Un esempio del possibile output è il seguente:

```
Somma degli elementi nell'intervallo [0;999] = 50063
Somma degli elementi nell'intervallo [3000;3999] = 48087
Somma degli elementi nell'intervallo [1000;1999] = 50270
Somma degli elementi nell'intervallo [2000;2999] = 49977
Somma degli elementi nell'intervallo [4000;4999] = 49250
Somma degli elementi nell'intervallo [5000;5999] = 50335
Somma degli elementi nell'intervallo [6000;6999] = 49346
Somma degli elementi nell'intervallo [7000;7999] = 48581
Somma degli elementi nell'intervallo [8000;8999] = 48823
Somma degli elementi nell'intervallo [9000;9999] = 49327
```