Programmazione a Oggetti

Esercizio sui thread

Si implementi un programma in Java che elabora un array di float di **N** elementi contenente valori casuali da [0,1).

L'elaborazione consiste nell'assegnare parti dell'array a **M** threads e definire un **range (vMin, vMax)** comune a tutti i thread, cosicché ciascuno di loro sia in grado di determinare se ciascun elemento della parte di array assegnato cada nel range definito.
In caso negativo, cioè se l'elemento è minore di vMin o maggiore di vMax, il thread deve incrementare un **Contatore tramite mutua esclusione** e quindi **sincronizzazione appropriata**.
Al termine dell'esecuzione dei thread, verificare il risultato della propria implementazione multi-thread con un metodo che passa in rassegna in maniera sequenziale l'intero array iniziale.

Si assuma che N > M, e che il lavoro sia bilanciato tra i thread.
In particolare, se M divide perfettamente N, allora ciascun thread avrà lo stesso numero di elementi da processare; diversamente, si pensi ad una strategia per allocare gli elementi "residui".

Si provi quindi ad testare la propria implementazione con N = 1024 ed M = 8.

Note:
Per generare numeri casuali è possibile usare la classe `java.util.Random`

```
//Istanza generatore di numeri casuali
Random rnd = new Random();
//generazione di numero float random in [0,1)
float numeroFloat = rnd.nextFloat();
```