Technische Hochschule Ingolstadt
Fakultät EI
Prof. Dr. Christian Facchi

Übung Verteilte Systeme
WS 2018/19
Übungsblatt 4

1) Aufgabe (threads in JAVA):

Im Folgenden soll ein typisches Schreiber/Leser-Problem realisiert werden. Verwenden Sie als Ausgangsbasis die jeweilig angegebenen Module. Es soll eine Schachfigur auf der Hauptdiagonalen wahlfrei zufällig verschoben werden. Hierzu werden zwei Threads zum Schreiben und auch zum Lesen der Position realisiert. Anmerkung: In den ersten Teilaufgaben erfolgt keine Synchronisation der Prozesse.

Gehen Sie hierbei wie folgt schrittweise vor.

a) Verwenden Sie die vorgegebenen Klassen `Figur`, `Schreiber`, `Leser` und `MachMal`. Erweitern Sie die Klasse Figur indem Sie die Methoden `setPosition` und `getPosition` (gibt die Position an stdout aus) in einer getrennten Klasse implementieren. Verwenden Sie hierbei Vererbung. Beim Setzen der Position bauen Sie eine Verzögerung von einer Zehntelsekunde (`MachMal.eineZehntelSekundeLangGarNichts()`) zwischen dem Belegen des X-Wertes und des Y-Wertes ein. Damit die Effekte auch beobachtbar sind, ergänzen Sie auch die Methode `getPosition` ebenfalls einer Verzögerung von einer Sekunde. Erstellen Sie nun die Klasse `FigurenThreads`, in der jeweils der Leser und der Schreiber gestartet werden. Achten Sie darauf, dass der Schreib-Thread im Hintergrund läuft. Dies erreicht man durch `setDaemon(true)`.

b) Starten Sie das Programm und interpretieren Sie die Ausgaben. Was ist die Erklärung für dieses Verhalten?

c) Synchronisieren Sie die Module.

2) Aufgabe (Synchronisation von Threads in JAVA):

Erzeugen Sie eine Klasse `UniqueId`, mit folgenden Eigenschaften:

- Dem Konstruktor wird als Argument ein Dateiname übergeben.
- Erstellen Sie eine Methode `init(<wert>)`, die den ihr übergebenen Wert als ganze Zahl in die mit dem Konstruktor angegebene Datei schreibt.
- Erstellen Sie eine Methode `getNext()`, die den aktuellen Wert aus der Datei liest und die Datei schließt um eine längere Verarbeitung zu simulieren. Anschließend wird der um eins erhöhte Wert in die erneut geöffnete Datei geschrieben und als Rückgabewert an die aufrufende Methode zurückgegeben.
- Verwenden Sie zum Schreiben und Lesen die Methoden (`readInt` und `writeInt`) der Klassen `DataOutputStream` und `DataInputStream` .
- Achten Sie hierbei darauf, dass die Methoden `getNext()` synchronisiert werden, da sie nebenläufig aufgerufen werden können.
- Erzeugen Sie nun eine Klasse `Test`, bei der:
- eine Datei `id.dat` erzeugt und mit einem Wert 10000 initialisiert wird
- jeweils 5 gleichartige Threads, also unterschiedliche Objekte, erzeugt werden.

- jeder dieser Threads 10-mal die ID mit der Methode `getNext` erhöht und seinen Namen, sowie den Rückgabewert ausgibt.

Testen Sie die Klassen ausreichend. Überprüfen Sie hierbei insbesondere die Programmausgaben.

Hinweis: Bitte beachten Sie dass der Scheduling-Algorithmus der JAVA-VM nicht näher spezifiziert ist. Insbesondere wird keine Fairness zugesichert.

3) Aufgabe (Synchronisation mit JAVA)

Gegeben ist die Klasse `IncDecThreads` als Vorlage, bei der jeweils ein Zähler von einem Thread erhöht und von einem anderen Thread nebenläufig erniedrigt wird. Beide Threads werden genauso oft durchlaufen.

- analysieren Sie das Programm.
- lassen Sie das Programm ablaufen und interpretieren Sie die Ausgaben
- sorgen Sie für eine geeignete Synchronisation, indem Sie das Schlüsselwort `synchronized` verwenden. Wieso funktioniert die naheliegende Lösung nicht?