Babeș-Bolyai Universität
Cluj Napoca
Fakultät für Mathematik und Informatik
Fortgeschrittene Programmierungsmethoden

Seminar 8

1. Implementieren Sie ein Java-Programm, das mehrere Worker-Threads erstellt, um Zahlen zu verarbeiten und die Ergebnisse anzuzeigen. Das Programm fragt den Benutzer nach der Anzahl der Worker-Threads. Es erstellt zwei Warteschlangen (input und output), um Zahlen und ihre verarbeiteten Ergebnisse zu speichern. Jeder Worker-Thread liest eine Zahl aus der input-Warteschlange, führt eine einfache Operation (zum Beispiel das Quadrieren der Zahl) durch und legt das Ergebnis in die output-Warteschlange. Ein einzelner Thread liest kontinuierlich aus der output-Warteschlange und zeigt die Ergebnisse an.

Erwartete Ausgabe (Beispiel):

```
Enter number of worker threads: 3
Thread-1 dequed 2
Thread-2 dequed 3
Thread-0 dequed 1
Thread-1 dequed 4
Thread-2 dequed 5
Thread-0 dequed 6
Thread-1 dequed 7
Thread-2 dequed 8
Thread-0 dequed 9
Thread-1 dequed 10
Result: 4
Result: 9
Result: 1
Result: 16
Result: 25
Result: 36
Result: 49
Result: 64
Result: 81
Result: 100
```

2. Implementieren Sie ein Programm, bei dem zwei Threads gleichzeitig einen Zähler erhöhen. Stellen Sie im ersten Teil der Übung sicher, dass aufgrund einer falschen Handhabung gemeinsam genutzter Ressourcen eine Racebedingung auftritt. Verwenden Sie dann im zweiten Teil eine atomare (Atomic) Variable, um die Race-Bedingung zu verhindern und die Thread-Sicherheit sicherzustellen. Zum Schluss implementieren Sie es erneut mit der Semaphore-Implementierung von Java.

3. Finden Sie heraus, was mit der folgenden Implementierung von Singleton nicht stimmt. Beachten Sie, dass es in einem Multithread-Kontext verwendet wird. Ändern Sie es so, dass es in einem Multithread-Kontext funktioniert

```
public class Singleton {
    private static Singleton instance = null;
    private Singleton() {}

    public static Singleton getInstance() {
        if (instance == null) {
            instance = new Singleton();
        }
        return instance;
    }
}
```